Shaping 3-D Volumes in Immersive Virtual Environments

Stenholt, Rasmus

Publication date:
2014

Document Version
Peer reviewed version

A Ph.D. dissertation by

Rasmus Stenholt

Shaping 3-D Volumes in Immersive Virtual Environments

AALBORG UNIVERSITY
DENMARK

The Faculty of Engineering and Science, Aalborg University, 2012

En Ph.D.-afhandling af

Rasmus Stenholt

Udformning af Rumlige Figurer i Immersive Virtuelle Miljøer

AALBORG UNIVERSITET

Det Teknisk-Naturvidenskabelige Fakultet, Aalborg Universitet, 2012

Shaping 3-D Volumes in Immersive Virtual Environments
Ph.D. dissertation

ISBN: 978-87-992732-7-0
Submitted: January 2012
Defended: May 2012
Printed: November 2014

Contents

List of Figures

List of Tables

List of Abbreviations

H_0	Null hypothesis
H_A	Alternative hypothesis
3C	3-Corner (Box Representation and Derived Box Shaping Techniques)
AABB	Axis-Aligned Bounding Box
ANOVA	Analysis of Variance
CAVE	CAVE Automatic Virtual Environment
CVA	Constant Visual Angle
DoF	Degree(s)-of-Freedom
FoV	Field-of-View
HIM	Hand-in-Middle technique
HMD	Head-Mounted Display
HOC	Hand-on-Corner technique
HRTF	Head-Related Transfer Function
LS-MOS	Large-Scale Multiple Object Selection
MOS	Multiple Object Selection (also sometimes called *volumetric selection* or *group selection*)
OBB	Oriented Bounding Box
PORT	Pointer Orientation-Based Resize Technique
RCBD	Randomized Complete Block Design
RIS	Real-time Interactive System
SOS	Single Object Selection
SS-MOS	Small-Scale Multiple Object Selection
TC	Two Corners technique

VDM Visual Data Mining

VE Virtual Environment

VR Virtual Reality

VW Virtual World

WIMP The Windows, Icons, Menus, and Pointers interaction paradigm

Dedicated to Louise, Ingeborg, and Holger

- *Rasmus*

Preface and Acknowledgements

This thesis is the result of many months of hard work. It presents a series of three papers, all somehow related to the common topic of shaping volumes of space. This work has presented me with many interesting and diversified challenges. The process has involved highly pragmatic issues such as collecting all the bits and pieces needed for building a small virtual reality laboratory. Then the troubles of making those bits and pieces work together. Of course, there have also been quite a few intellectual challenges best solved in using equal measures of imagination and pen-and-paper. During this process, many people have given advice, or otherwise assisted me. Wanting to avoid the risk of accidentally forgetting anyone, I would like to spend this opportunity to make a collective statement of gratitude to everybody who participated in some way, whether small or large, direct or indirect.

Reader's Guide

The thesis is split into two overall parts: The main report, and the actual collection of contributed papers. The aim of the main report is to present the contents of three papers in a coherent fashion. It contains an introduction, a methodology chapter, an extended summary of the papers, and a conclusion. Before the summary of the papers, a technical chapter on the geometry of boxes is presented. The main report ends with a bibliography. Likewise, each of the three contributed papers have their own bibliographies. References to the bibliographies are presented in square brackets, e.g. [6]. The bibliographies are sorted alphabetically by surname of the primary author. A list of the abbreviations used is found alongside the lists of figures and tables at the beginning of the thesis.

Some of the papers and chapters present mathematical topics. In general, the mathematical notation used is as follows: Scalars in lower-case italics, e.g. s. Points in upper-case italics, e.g. P. Vectors, and quaternions in bold, lower-case italics, e.g. $\boldsymbol{v}$, and $\boldsymbol{q}$. If a point has been turned into a vector, e.g. for use in a matrix transfomation, the notation follows the vector typeface. Matrices in bold upper-case italics, e.g. $\boldsymbol{M}$. A superscript ^{T} is used to denote the matrix/vector transposition operator, e.g. $[1,0,0]^T$ is the transpose (a column vector) of the row vector $[1,0,0]$, and $\boldsymbol{M}^T$ is the transpose of the matrix $\boldsymbol{M}$.

Abstract

Shaping 3-D volumes is an important part of many interactions in immersive virtual environments. The range of possible applications is wide. For instance, the ability to select objects in virtual environments is very often based on defining and controlling a selection volume. This is especially true, if the intention is to select multiple objects. Another important application area is the manipulation of objects through the use of controllable handles, or widgets. Such widgets are often associated with a bounding volume around the object to be manipulated. Such techniques are both well-known in 2-D and 3-D contexts. A third application area is the task of 3-D modelling. In this case, the shaping of simple 3-D volumes is an intrinsic part of building a complex 3-D model from primitive shapes such as boxes, spheres, and cylinders.

This thesis presents three separate, but sequential, papers within the area of volume shaping. As such, each paper builds on the results of its predecessors. The first paper deals specifically with the task of shaping a 3-D box. Boxes are arguably one of the most commonly used 3-D shapes, both in real life and in virtual worlds. Therefore, the problem of efficiently and precisely defining a 3-D box is a fundamental one to investigate. The first paper does this by analysing the practical task of defining a 3-D box as the equivalent task of defining its degrees-of-freedom. This analysis leads to the introduction of a new way of shaping a box from just three of its corners organized in a specific configuration. Subsequently, box shaping techniques based on this idea are formally evaluated in a user study to be significantly more precise and no slower than existing techniques.

The second paper focuses on exploration and perception of structures in 3-D scatterplots. In data mining contexts, being able to perceive the shape of a structure is a prerequisite for making precise subsequent interactions such as selections and annotations. The paper introduces a new type of 3-D glyph rendering, which potentially diminishes the negative effects of clutter while reinforcing the perception of shapes. This new glyph type is evaluated against regular glyphs under various experimental circumstances, and is found to improve perception of structures and shapes in cluttered conditions. Furthermore, two volumetric density thresholds are discovered. Showing 3-D scatterplots with densities above/below these thresholds significantly changes the shape perception performance of users.

In the third and final paper, volume shaping is used in the context of efficient selection of multiple objects. The paper introduces a new technique, the magic wand, which automates a large part of the user's work in such tasks. This technique is compared to two other techniques, a spherical brush and a box-shaped

lasso, in an evaluation which seeks to identify the pros and cons of the tools. The magic wand proves to be faster to use than the other, but only in certain geometric scenarios. The brush, on the other hand, performs decently across most of the tested scenarios. The box lasso was judged by the users to be difficult to use compared to the other techniques, even when the geometric scenario was perfectly fit for a box-shaped tool.

Synopsis

Udformning af 3-D former er en vigtig bestanddel af en lang række interaktionsteknikker i virtuelle miljøer. Udvalget af anvendelsesområder er meget bredt. For eksempel er teknikker til at løse selektionsopgaver, dvs. opgaver hvor målet er at udvælge et antal virtuelle objekter, meget ofte baseret på at definere og styre et såkaldt selektionsvolumen. Dette gør sig især gældende, når målet er at udvælge flere objekter med én operation. En anden vigtig anvendelse er manipulation, fx forandring af placering, orientering og form, af virtuelle objekter ved hjælp af små, virtuelle håndtag. Sådanne håndtag er ofte knyttet til hjørner og flader på et volumen, der omslutter det objekt, som skal manipuleres. Dette er et velkendt interaktionsprincip både i 2-D og 3-D. Et tredje eksempel på en anvendelse er 3-D modellering. I dette tilfælde er udformningen af komplekse 3-D modeller ofte baseret på skabelse og manipulation af mange simplere rumlige former såsom kasser, kugler og cylindre.

Denne afhandling præsenterer tre særskilte artikler omhandlende forskellige aspekter af at udforme rumlige figurer i virtuelle miljøer. Artiklerne bygger sekventielt på hinandens resultater. Den første af disse artikler omhandler udformningen af kasser i 3-D. Kasser og kasselignende former er utvivlsomt en af de mest anvendte og velkendte klasser af rumlige figurer - både i virkelighedens verden og i virtuelle miljøer. Derfor er det at kunne skabe kasser effektivt og præcist et grundlæggende problem at undersøge. Den første artikel angriber dette problem ved at analysere det konkrete problem at skabe en kasse som de ækvivalente problem at definere en kasses frihedsgrader. Denne analyse fører til introduktionen af en ny måde at udforme en kasse ved blot at udpege tre af dens hjørner i et bestemt mønster. Dernæst evalueres disse teknikker i et formelt brugerstudie, og findes at være mere præcise end eksisterende teknikker, samtidig med, at de ikke er langsommere.

Den anden artikel omhandler udforskning og perception af strukturer og former i 3-D plot af samhørende værdier. I eksplorativ dataanalyse er det at kunne opfatte større strukturer præcist en forudsætning for at kunne lave efterfølgende præcise selektioner og annoteringer. Artiklen introducerer en ny måde at fremvise 3-D punkter i sådanne plot. Denne fremvisningsmetode kan potentielt reducere de negative virkninger af uønskede datapunkter, som blokerer for et frit udsyn til det ønskede struktur, samtidig med at større strukturer bliver fremhævet. Metoden sammenlignes i et formelt brugerstudie med den traditionelle fremvisningsmetode, og findes at være signifikant bedre, når der er mange uønskede datapunkter. Desuden dokmenteres det også, at der findes to grænseværdier i den rumlige tæthed af viste datapunkter. Ved højere tæthed end disse græn-

seværdier, forøges evnen til præcist at opfatte og skelne mønstre væsentligt.

I den tredje og sidste artikel bliver udformning af rumlige figurer brugt i en konkret anvendelse, nemlig effektiv selektion af flere objekter. Artiklen introducerer en ny teknik, magic wand, som automatiserer en stor del af selektionsprocessen. Denne teknik sammenlignes med to eksisterende teknikker, en kugleformet pensel og en kasseformet lasso for at identificere fordele og ulemper ved de forskellige teknikker, når de anvendes i en række forskellige testscenarier. Teknikken magic wand var signifikant hurtigere end de andre to teknikker i visse scenarier. Til gengæld var penslen et godt kompromis på tværs af alle scenarier. Den kasseformede lasso blev bedømt som en svær teknik at anvende, selv når scenariet præsenterede en kasseformet sky af datapunkter, som skulle udvælges.

1 Introduction

1.1 Virtual Reality, Virtual Environments, and this Thesis

The concept of *virtual reality* (VR), i.e. replacing the stimuli of the real world with artificially generated stimuli in a realistic and convincing manner, is a challenge in many ways. Many recent advances in technology has made VR a feasible and realistic concept with a wide range of possible applications and implementations. The main VR challenges fall into several categories. In this introduction, we take a brief look at each of these categories in turn, to explain where the body of work presented in this thesis fits into the big picture of VR.

Hardware for Virtual Environments

A common prerequisite of any kind of VR system is the hardware needed to present the desired stimuli. The challenge in this respect is to make the hardware powerful and responsive enough to convincingly deliver the content of the virtual world (VW) while remaining unobtrusive and easy to use.

All VR hardware fall into either of the two main categories of input and output hardware. In this thesis, a complete setup consisting of all the needed input and output hardware will be referred to as a virtual environment (VE).

One example of hardware, which features prominently throughout this thesis, is a motion-tracked helmet with two integrated displays, i.e. a head-mounted display (HMD). The integration of motion tracking and two separate displays potentially provides a very convincing visual experience, which supports the sensation of depth both through stereopsis and motion parallax. The first motion-tracked head-mounted display was introduced as early as 1968 by Sutherland [55]. When a stereo display system is combined with head tracking, we choose to refer to such a VE as an immersive VE.

A quite different approach to VE's is known as a CAVE, in which the user is surrounded by a number of projection screens, usually 3-6. In the case of 6 screens, the user becomes completely contained inside a display cube, allowing for viewing of the VW from any desired viewpoint. Being introduced in 1992 by Cruz-Neira et al [10], the CAVE is a more recent invention than the HMD. The CAVE and the HMD can both present depth information through stereo images, however, in a CAVE this effect is accomplished through the use of 3-D glasses capable of correctly separating the left and right eye images shown on the pro-

jection screens. Other VR display systems also exist such as Fish Tank VR [68], however, most immersive VE displays somehow follow either the HMD or the CAVE paradigm.

As is the case with an HMD, a CAVE also needs real-time tracking of the user's motions to function. Indeed, motion tracking is a prerequisite technology of almost any VE, making it a very important category of input hardware in VR systems. There are many approaches to motion tracking for VR purposes, e.g. mechanical, ultrasonic, magnetic, optical, and inertial tracking. Each type of system has distinct advantages and disadvantages as e.g. presented in [69]. For instance, the system employed in the experiments of this thesis is an infrared optical system[1], which has the advantages of being cheap and easy to expand in terms of the number of tracked objects. The drawback is mainly that the tracked objects must be visible at near-infrared wavelengths, making optical systems prone to visual occlusion problems.

The system delivering the images to the display system is very often a powerful PC, or a cluster of such PC's, with high-end accelerated 3-D graphics cards. A PC also forms the heart of the system in the experiments of this thesis. This computer has the task of receiving and interpreting input from the optical tracking system and the interaction devices, rendering an image that matches the current situation, and delivering this image to the head-mounted display in response to the user's actions. A 3-D rendering of the laboratory environment used is shown in Figure 1.1.

Although vision is the sense most commonly stimulated by a VE, it is not the only one. Probably the second most commonly stimulated sense in VE's is the auditory sense. As is the case with real-time 3-D computer graphics for the visual sense, very convincing technologies for real-time rendering of 3-D audio exist. They mainly fall into either of two categories, depending on whether they depend on headphones (binaural technologies), or on arrays of loudspeakers. A prime example of a state-of-the-art binaural technology is called head-related transfer functions (HRTF) [20, 75]. Examples of technologies based on speaker arrays are Wave-Field Synthesis [2] and Ambisonics [17]. Although the lab used in the experiments of this thesis is equipped with a speaker array capable of rendering Ambisonics spatialized audio, the only output modality used in the studies is visual feedback. This choice was made to simplify the studies. However, future studies integrating audio and visuals into the same experiments is definitely an interesting prospect.

The sense of touch can also be stimulated through the use of various haptic and tactile feedback devices such as gloves and joysticks. While the sense of touch is definitely an important part of a realistic environment, we have chosen to focus only on visual feedback. This choice was made for several reasons: Haptic technology seems somewhat immature compared to auditory and visual feedback technology. Furthermore, no haptic equipment or expertise was available in-house. The same can be said of olfactory and gustatory feedback, which furthermore are expected to have little relevance or consequence when it comes

[1] Specifically, a 24-camera OptiTrack setup is used. See `http://www.naturalpoint.com/optitrack` for more information.

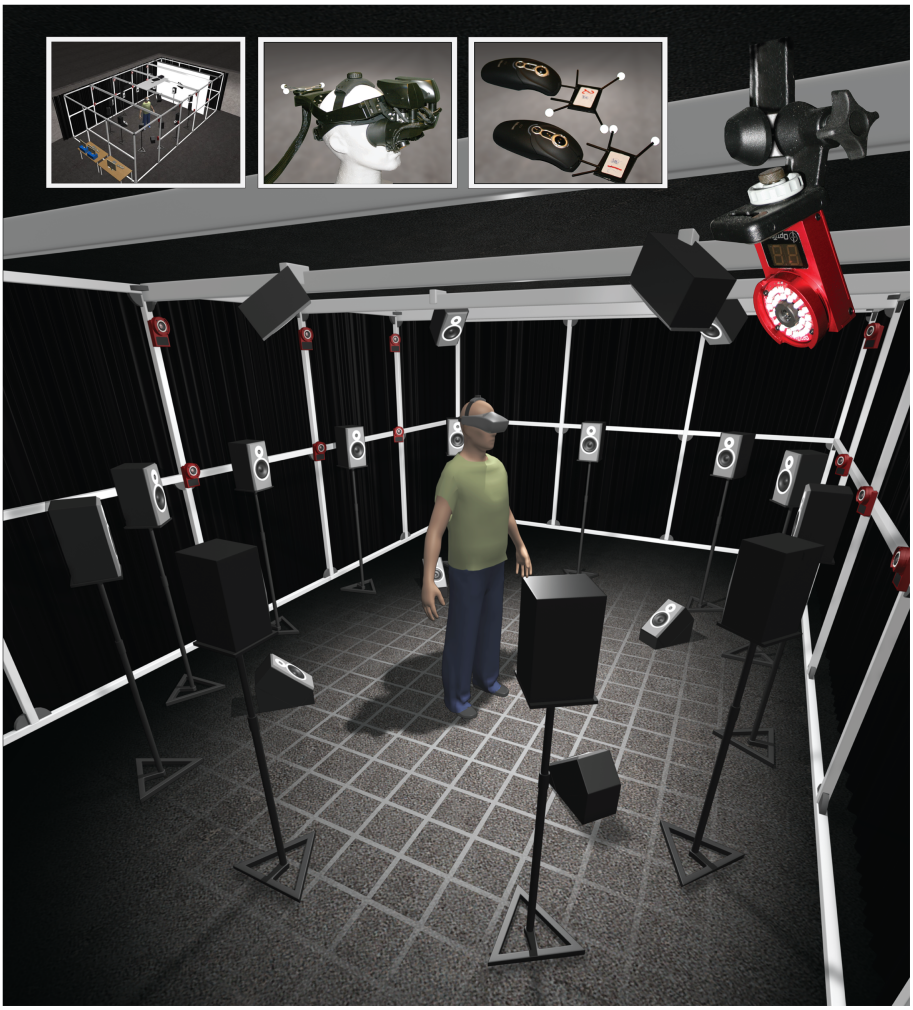

Figure 1.1: A 3-D rendering of the laboratory environment used in all experiments of this thesis. The user is standing in the middle of a large aluminium frame with infrared motion tracking cameras mounted at the periphery of a $4.5m \times 4.5m$ floor space. The cameras show as the small, red objects in the image. The illustration also shows the speaker array present in the laboratory. The three inserts at the top of the figure show a bird's eye view of the lab, a photograph of the lab's HMD, and the interaction devices used.

to users' ability to perform the task that is the main topic of this thesis: Shaping 3-D volumes.

Apart from the navigational options offered by head tracking, a common type of input requirement in most immersive systems is the ability of the user to interact with the VW. The range of devices available for this purpose is wide and varied. Some common categories of input devices for use in immersive contexts are glove-type devices and wand-type devices. These categories of devices are especially useful for interactions involving grabbing and pointing, respectively. In this thesis, the typical user interaction needed to achieve the goal of volume definition is essentially a pointing task. For this reason, wand type devices are used. There are many other kinds of devices besides wands and gloves, however, these devices are typically highly specialized for a specific application. Some examples of such devices are presented in [22, 15].

Software for Virtual Environments

As was the case with hardware, the software systems used for rendering the virtual worlds is also somewhat specialized. Such systems are often referred to as real-time interactive systems (RIS). The typical requirements of RIS software framework are challenging to integrate into a single package:

1. Interfacing with numerous specialized input and output devices.
2. Integrating motion tracking information from head and hand motions.
3. Real-time, low latency 3-D graphics rendering.
4. 3-D spatialized audio rendering.
5. 3-D scene management and model import/export
6. Collision detection and physics simulation.
7. Handling of distributed and parallel computing, with associated concurrency issues.

Many systems adressing various subsets of these tasks exist. For instance, the OpenSG scene graph is designed for rendering 3-D graphics with support for immersive VE's [47]. A small overview of the topic along with a presentation of a specific system called Simulator X is presented in [30]. VR Juggler [3] is another example of a general VR software framework.

In this thesis, the requirements of the experiments made are both quite specific and limited compared to the functionality offered by full-fledged VR software systems. The main requirements is the ability to receive tracking information from the OptiTrack system and the employed interaction devices, and to render the correct graphics in response to user actions. Thus, there is no need for physics simulation, distributed computing, audio rendering etc. For this reason, a simple, custom-made software platform is used in all experiments. This platform supports exactly the functionality needed, and not much else. The platform is based on OpenGL+GLUT for producing the 3-D graphics and NaturalPoint's Tracking Tools API for motion tracking using OptiTrack.

Interaction Techniques for Virtual Environments

With the hardware and software for the VE in place, the topic of interactions in the virtual environment comes next. The topic of this thesis is concerned with some specific challenges posed by such interactions. Specific ways of solving particular interaction tasks are often referred to as *interaction techniques*. Some of the interaction techniques for virtual environments have established themselves as accepted standards for solving specific tasks. E.g. the laser beam metaphor found in the ray casting technique [37] used for pointing at distant objects is very common for that particular purpose. It is worth noting, though, that many variations on the basic techniques exist. Each of these variations address specific shortcomings of the original version, e.g. pointing at objects in cluttered environments [11]. Another type of variation on basic interaction techniques is based on using novel devices for input, typically offering new affordances and functionality, to improve and modify existing techniques.

However, no single 3-D interaction technique, or set of such techniques and devices, has gained the same universal paradigm status as the windows, icons, menus, and pointers (WIMP) paradigm [62] controlled by a keyboard and a mouse for 2-D desktop use, or the multitouch paradigm used in smart phones and tablet computers. There are also many tasks in VE's where no generally accepted standard way of solving that task exists. One such task is the simultaneous selection of multiple objects, which is the focus of one of the papers presented in this thesis.

Compared to 2-D user interaction, 3-D user interaction is both complicated by the lack of standards, and by the increase in the number of degrees-of-freedom (DoF) that a user must control to accomplish any task. The notion that any interaction task is subject to a number of DoF, which correlates with the task complexity, plays a central role throughout this thesis. The prime example of this is the first paper, which investigates the allocation of user control to the DoF of a rectangular 3-D box.

Another problem which arises often in 3-D contexts is visual occlusion. The 3-D nature of the VW, means that objects of interest may be occluded by other objects, depending on the user's viewing position. In 2-D contexts, this problem is alleviated by the fixed position of the user relative to the content on the screen, which allows the content of interest to be presented such that none of it is occluded by other content. This problem is approached in the second paper presented in this thesis.

Interaction techniques for use in VE's have typically been divided into 4 categories, depending on their purpose in the VW [6]:

- Navigation
- Selection
- Manipulation
- System control

This distinction of categories is general enough to capture most conceivable interaction tasks. However, many subdivisions exist within each category in or-

der to address specific problems and applications. The split between selection techniques for single objects and multiple objects, as studied in the last paper of this thesis, is one example of such a subdivision. A brief overview of each category is given, in order to place the contributions of this thesis in relation to these four categories:

Navigation

Navigation covers all interactions related to moving around in a virtual world. As such the navigation category covers several aspects of moving in a virtual space, e.g. both the planning of future movements, the mental mapping of a space, as well as the actual performance of the movements. Many techniques and hardware interfaces exist which support navigation of a virtual space.

One of the most common and natural ways of navigating a virtual space is to walk. The walking can either be a direct, natural walk as in the real world, or be aided by devices like treadmills or special floor mats. Lots of studies about walking in virtual spaces have been made. Recent example are e.g. found in [64, 54]. Indirect walking can e.g. happen by pushing a joystick, or by simulating a walk by tapping two fingers on a touch sensitive surface [25]. Viewed at a general level, control of a virtual walk is typically constrained by gravity to 2 translational DoF and 1 rotational DoF around the axis of gravity.

Another commonly used navigation metaphor is the flight metaphor, where the user is no longer bound to the virtual ground by gravity. Flying is often more complicated to control than walking, because completely unconstrained flight allows vertical translation as well as rotation about three axes, i.e. a total of 6 DoF. Another navigation metaphor inspired by a real-world transportation form is driving. Techniques based on completely unnatural navigation metaphors also exist, such as teleportation through pointing on a map of the environment.

The navigation performed in the experiments of this thesis only allows for natural walking in a space constrained by walls and the length of the cable attached to the HMD. Therefore, the thesis contributes no new navigation techniques. However, the second paper presents a way of rendering objects in a virtual space, such that navigating the space changes the perceived size of the objects.

Selection

Selection is the action of somehow picking one or more objects in a virtual world. Selected objects can then be the subjects of subsequent interactions. Most of the current work on selection in VE's deals with the problem of selecting a single object at a time. As mentioned previously, one of the common ways of selecting a single object is through the use of a virtual laser beam shining outwards from the user's hand. Another approach is to allow the arms of the user to stretch unnaturally into the virtual world, until the virtual hands touch the desired object. The most well-known example of such arm-stretching techniques is probably the Go-Go technique presented in [46].

There are also examples of more indirect techniques, where the selection is not based on directly touching or intersecting the desired object. This can e.g. be achieved by picking the object from a list of object names or icons. One recent instantiation of this idea combines an initial, direct ray casting step to pick a group of objects, whereupon the specific object in the group is selected by going though a series of iconic menus representing the objects [28].

Being able to select multiple objects is one of the main application areas of shaping a volume of space. Most of the current works on the topic use 3-D selection boxes [60, 58]. This is an approach, which also plays an important role in all of the contributions of this thesis. Thus, the selection category, specifically the one concerning multiple objects, is one where the contributions of this thesis can potentially provide an improvement of the state-of-the-art.

Manipulation

Traditionally, manipulation interactions are those that modify the position and orientation of an object. In a broader sense, manipulation can also be thought of as any interaction that somehow modifies 3-D objects. Manipulation techniques are very often closely tied to the selection scheme. I.e. it is often the case that selection and manipulation techniques are combined into one general technique for handling both categories of interactions. One example of such an integrated, or hybrid, technique is HOMER [5]. Manipulation techniques that only modify the position and orientation of objects require 6 DoF of control. Manipulation techniques for changing the scale of an object also exist, e.g the PORT technique presented in [33].

Defining a volume of space can also be used in object manipulation. The general idea in this approach is that the created volume of space has some number of DoF, depending on the geometric shape of the volume. These volume DoF can in turn be mapped to the controls of the DoF of an object to be manipulated. A well-known example of this from the realm of desktop computing, is the use of bounding rectangles, or widgets, to rescale objects. The box shaping techniques presented in the first contributed paper of this thesis would e.g. be applicable as a manipulation technique for 3-D contexts in this way.

System Control

The final category of interactions is system control, which covers any interaction controlling parameters outside of the virtual world, or controlling things which are not directly represented by an object in the virtual world. As such, system control represents everything which is not inherently navigation, selection, or manipulation. Examples of system control interactions would be to change the simulation speed, to pick a different navigation mode, to choose another selection tool, to save a model of the world, etc. In desktop environments, system control options are often controlled through the use of menus and dialogue boxes. However, in immersive VE's, presenting this information in an intuitive and non-obstructive way is difficult. In this thesis, all system control aspects will either be controlled automatically by the expermental software, e.g. moving on to the next

test condition, or manually by the experimenter, who is not immersed in the VE. Thus, no new contributions are given in the system control category.

1.2 Motivation

In the above introduction, the relationship between VR and VE's in general and this thesis has been established. In this section, the goal is to motivate why the definition of volumes of space is an important and interesting topic in the broad field of 3-D user interactions. We will do this through a few examples. Some of these examples have already been briefly mentioned previously.

Interactive Modelling in Immersive Environments

In some cases, the use of an immersive VE provides new and interesting opportunities for 3-D sculpting and modelling. This can e.g. be the case in scenarios, where it is important to experience the correct scale of the model during the modelling proces. The usefulness of defining a volume of space in this context is not restricted to the detailed modelling process. It can also be used in describing exactly where and how to introduce premade objects into the virtual scene. This may be very useful for applications like architectural design. In that case, the use of a volume definition interaction becomes closely linked to the manipulation category of 3-D user interaction. Another application area, where modelling may be useful is in game-like contexts, where the entire point of the game may be the construction of models, much like a set of Lego bricks. A recent, popular example of such a computer game from the desktop world is Minecraft[2].

Selection of Multiple Objects

Multiple object selection (MOS) is a very large and important application area of immersive volume definitions. Being able to efficiently select many objects is the prerequisite of other interactions with multiple objects. In this case, the volume defined by the user is used as a selection (or deselection) volume, within which all objects are affected in the same way. This volume can be shaped in many different ways. A selection volume can e.g. be defined by dragging a persistent, primitive shape around to create a complex, swept volume from the union of all the places that the primitive shape has been. Another approach is to define a specific type of volume, e.g. a box or a cylinder, from scratch using an efficient, tailor-made construction technique for that particular shape. Shaping volumes of 3-D space for MOS purposes is the subject of study in the final paper presented in this thesis. Examples of subsequent operations following the MOS are e.g. annotation, zooming in on a volume of interest, cloning of selected objects, or putting the selection targets in a different place in the virtual world.

[2]See `http://www.minecraft.net` for more information.

High DoF Manipulation

If the volume of space defined by the user has more than the usual 6 rigid transformation DoF coming from translation and rotation, the option to use the extra volume DoF to control the DoF of the object(s) inside the volume presents itself. We choose to call this *high DoF*, because it goes beyond the usual rigid manipulations for which a multitude of efficient and appropriate techniques already exist. For instance, non-rigid geometric transformations such as scale, shear, and reflection can be controlled by moving the vertices, edges, and faces of a bounding object of an appropriate number of DoF. Thus, the shaping of a volume of space becomes a manipulation technique in its own right. This approach to manipulation is often referred to as widget-based manipulation.

1.3 Thesis Outline

This thesis consists of the following main chapters and contributions. At first, the introduction presented in chapter 1 seeks to introduce the topics of this thesis in relation to the broad field of VR. This chapter also seeks to motivate the topic of the thesis through three different examples of possible applications of immersive volume definition.

The next chapter, chapter 2, presents the considerations made about the overall methodology of the studies presented. This chapter goes into detail with the overall approach to scientific investigation chosen, as well as how that approach has been implemented in the studies presented in the contributed papers.

The methodology chapter is followed by a chapter about the geometry of boxes, chapter 3, and how to represent them in a pratical way, both for user interactions and computational purposes. This chapter is included since it provides a lot of the mathematical background knowledge, which is later distilled into the box shaping techniques employed in all the papers presented in this thesis.

Next, a summary of the contributions made in relation to the current state-of-the-art is presented in chapter 4. This chapter outlines the contents of all three papers of the thesis and places them into the common framework of this thesis, the shaping of volumes of 3-D space. The final chapter presents the conclusion to the thesis along with some thoughts on the perspectives of the thesis and some suggestions for future work deriving from this thesis.

The final part of the thesis is the presentation of the three paper contributions, Paper A, Paper B, and Paper C.

2 Methodology

Overall, the three papers of this thesis are consistently based on the same method. This method encompasses everything from the design of the experiments and the measurements made, to the analysis of the experimental results. This chapter illuminates what this method is, how it is carried out, and what its philosophical underpinnings are.

2.1 Critical Rationalism, Hypotheses, and Statistics

The general approach to science taken in this thesis is that proposed by K. R. Popper as *critical rationalism* [44, 45]. Very briefly put, the critical rationalism view dictates that scientific investigation should seek to *falsify*, rather than *verify*, its hypotheses and theories. This implies that one can never verify hypotheses, but merely fail to falsify them. This view makes sense, since no amount of confirmation can rule out the odd possibility of finding a contradiction in the next observation or sample. Thus, if we find that no contradictory evidence currently exists, e.g. through experimentation, then we allow ourselves to state that the hypothesis/theory has not yet been falsified, instead of accepting the hypothesis as verified. Thus, any scientific experiment using critical rationalism as framework involves at least three steps:

1. Formulate a number of *falsifiable* hypotheses.
2. Devise and perform an experiment which can falsify the hypotheses.
3. Analyse the experimental results with the aim of falsifying the hypotheses.

The above steps are exactly those, which have been followed in all of the papers included in this thesis. The notion of using hypotheses and falsification as scientific framework is very closely tied to the ideas presented in the framework of statististical hypothesis testing. There, falsification is based on phrasing appropriate null and alternative hypotheses, H_0 and H_A, respectively. The goal of the statistical test is then to calculate how probable the given data is, provided that H_0 is true. If the data is in agreement with H_0, we allow ourselves to say that we have failed to reject (falsify in Popperian terms) H_0 at the chosen significance level. Otherwise, we choose to say that we cannot accept H_0 at the chosen level

of significance. The notion of statistical testing provides a very convenient and operational implementation of Popper's critical rationalism. 66

The significance level, α, can be arbitrarily chosen. A higher significance level, implies a lower the risk of falsely rejecting H_0. In fact, α represents the probability of committing such an error, called a type I error. The value of α, however, should not be chosen to be too close 0%, since this increases the risk of falsely failing to reject H_0, called a type II error. For this reason, the choice of significance level in all analyses made in this thesis has been chosen at $\alpha = 0.05$. I.e. a probability of 5% of making type I errors is deemed acceptable.

Thus, the experiments and measurements of this thesis must all be designed such that the ideas expressed in the critical rationalism approach to science can be followed and implemented in practice using statistics as a tool. A discussion of the practical consequences of this choice is presented in the following section.

2.2 Measurements

The main issue in designing experiments that allow for statistical hypothesis testing is in the types of measurements that are allowed. Statistical methods are only applicable, if the measurements adhere to certain rules. These rules and the choices made about measurements in the experiments are presented in this section.

Quantitative vs. Qualitative Data

Overall, there are two kinds of data that can be gained from an experiment: Quantitative and qualitative data. With quantitative data, each measurement is representable as a number. Such measurements dominate scientific disciplines such as physics and chemistry. Quantitative measurements are also frequently the output of questionnaires. In the context of this thesis, quantitative measurements can e.g. measure quantities like error size and count, completion time, and user preference. Quantitative data go hand in hand with statistical analysis, since all statistical analysis methods are based on being able to make statements in terms of numbers such as probabilities and uncertainties. This is the great strength of quantitative measurements.

The main weakness of quantitative measurements in a user study context is that they leave little possibility of explaining *why* the results come out the way that they do. Thus, they offer little or no option for the participants of an experiment to motivate their opinions, or express themselves in a high level of detail.

Qualitative methods, on the other hand, allow users to express their opinions more or less freely through interviews, video recordings, or written accounts of their experiences. This kind of data is very valuable in understanding why users have a specific opinion. However, no statistical analysis of the results is possible, unless the experimenter allows the results to be subjectively interpreted. As such, it is impossible to express the certainty of results in the case of qualitative measurements.

Because of the choice of the critical rationalism approach, we will deal only with quantitative results of experiments, such that the stated hypotheses can be

falsified with a chosen degree of certainty. However, users will in some cases be asked to quantify their opinions about the difficulty of the given task/trial. This kind of measurement represents a quantification of something which is otherwise inherently easiest to explain in qualitative terms.

Using a quantitative approach does not rule out getting qualitative feedback. Thus, participants are encouraged to freely make comments both during and after an experiment, which can be used in understanding the gathered quantitative data.

Objective vs. Subjective Data

Another dimension to consider when making measurements is whether the measurements are *subjective* or *objective*. A measurement is considered to be subjective, if it is based on the subjective opinion of the participant in the study. The prime example of such a measurements in this thesis, is the perception of task difficulty rated on a 1-10 scale. On the other hand, objective measurements are those, which are not based on information reported directly by the participants. The completion time of a task is an example of such a measurement.

It is important to note that a subjective measurement does not have to be qualitative. The types of measurements used in this thesis are all of an objective or subjective quantitative nature. This is also illustrated in Table 2.1.

Table 2.1: A small matrix illustrating the four types of measurements considered. The options marked by X are the only ones used in this thesis.

	Objective	**Subjective**
Quantitative	X	X
Qualitative		

Levels of Measurement

Within the realms of quantitative measurements, four so-called *levels of measurement* exist [52]. Moving through these four levels allows for an increasing number of statistical analyses to be made. The levels and their relation to the measurements of this thesis are:

1. **Nominal/categorical** measurements take values that fall into discrete categories represented by a set of labels. As such, categorical variables mathematically represent unordered sets. Categorical variables feature prominently as explanatory variables in all three studies. For instance, in one of the studies, the choice of selection technique is such a variable.

2. **Ordinal** measurements are those where the values can be ordered, making the median (but not the mean value) a meaningful quantity. I.e. they form an ordered set. In this thesis, all subjective measurements are essentially on ordinal scales. The rationale of this is that although a difficulty score of 2 is

definitely greater than a score of 1, it is not certain that it implies that the first task was twice as difficult as the second one.

3. **Interval** level measurements are ratio level variables with an arbitrary origin. I.e. the value 0 is not necessarily the point where there is no presence of the quantity measured. Thus, sums, differences, and mean values are meaningful, but ratios of measurements are meaningless. Interval level data are quite uncommon, and there are no such variables in this thesis.

4. **Ratio/continuous** measurements are those which typically take values from the line of real numbers. This is the most common type of measurement in natural sciences. Continuous variables are also the most versatile in terms of possible analyses. In this thesis, completion time and some error and quality measurements are made on ratio level scales.

One problem to consider related to levels of measurements is whether or not to allow ordinal variables to be treated by analyses otherwise designed for ratio level responses, e.g. analysis of variance (ANOVA). Analysis methods that rely on the properties of ratio level responses, like ANOVA or linear regression, are more powerful and offer greater explanatory possibilities through modelling of the data. This is the main reason for wanting to use these methods even on data, where they are theoretically not applicable. In this thesis, the pragmatic choice of using the analysis methods anyway has been taken. This is in line with common practice in many other studies, where data like Likert scale measurements are subjected to ANOVA analysis. A recent justification of this choice is presented in [43].

Measurements & User Tasks

In experiments involving a user performing some pre-defined tasks, some consideration must be made as to when a task is done. A few options immediately present themselves:

1. The task is done, when it has been solved to perfection, or acceptably close to perfection.

2. The task is done, when the user has performed some pre-defined number of steps, regardless of the quality of the end result.

3. The task is done, when the user is satisfied with the result.

4. The task is done, when some time limit expires (if not sooner by one of the other criteria)

The main point to observe about these options is that the first one is fundamentally different from the other options. This fundamental difference arises as a consequence of the requirement of perfection. The advantage of requiring perfect results is that the end result becomes very well-defined and predictable. As such, problems with outliers in subsequent analysis may be reduced. However,

if a perfect result is required, then it becomes essentially meaningless to measure the quality of the achieved result. This implies that the only meaningful measurements are related to the process of the task, and not the results. Examples of process-related response variables are e.g. completion times, measures of path length, or the number of operations used to achieve the result. Another implication is that all given user tasks must be perfectly solvable within a reasonable time frame to make the experiment practically feasible. This rules out experimental conditions, where the task is expected to be very difficult, or even impossible, to solve to perfection. The conditions for perfection can be relaxed somewhat by allowing a small margin of error. However, such an approach will still not illuminate how precise a given interaction technique is, or how large errors users are likely to make.

In all of the other cases, a task is not ended by a perfection criterion. Thus, the quality of the end result becomes a meaningful quantity to measure in addition to process-related measures. This is a great advantage in studies, where the main point is to test a number of interaction techniques both in terms of process and end results. However, the end results also become much less predictable. E.g in the case of ending the task based on user satisfaction, the user may choose to simply give up, or be happy with a poor result. Another group of users may be very demanding of their results, spending much time and effort before being satisfied. This fact is likely to cause great variability in the measured responses.

Since the studies of this thesis in all cases are interested in the end results of a user task, the choice is to go with the options that allow meaningful measurement of end result quality. However, the approach of setting an upper time limit on a trial will not be used. Instead, participants in the experiment will be given a practice session, such that some level of proficiency can be achieved prior to making the measurements. This should, at least partially, remove the problem of great variability in completion times.

The next choice about the user tasks is whether or not to allow the users to restart a trial. The approach of this thesis is to allow restarts, if the participant feels that a mistake has been made, e.g. by inadvertently clicking a button at the wrong time, or when a glitch in the motion tracking system occurs. The rationale of this approach is that it best mimics the conditions under which users would normally apply a technique. I.e. if you discover a mistake, you typically correct it instead of leaving it. In one of the studies, each user task is completed when all the steps of the tested interaction technique have been carried out. Some of the techniques tested only involve one step, which means that there is no opportunity to spot mistakes after completing that single step. However, in those cases, the results of completing the single step will be clearly presented prior to completing that step.

When considering the permission of restarts, it is also worth considering the amount of refinement/iteration allowed after the initial volume creation. The studies of this thesis handles this differently, depending on the goal of the experiment. E.g. in the experiment, where the precision of a specific volume definition technique is the subject of study, refinement of the initially constructed volume is not allowed, because these refinement steps are different from the actual volume definition technique. Thus, the addition of refinement steps in that case would

pollute the results regarding the precision of the initial volume definition. In other studies, however, where this is not an issue, an iterative scheme has been allowed.

Sources of Visualization Data

The final overarching methodological concern to address is the source of data to be used in the visualizations of the experiments. This data is needed in two of the experiments, which deal with volumetric pattern perception and multiple object selection, respectively. The main options are:

1. Real-world data
2. Simulated data
3. Synthetic data

Real-world data is data which has been collected from some real-world source. The great advantage of real-world data is that it immediately offers a practical application scenario of the tested interaction techniques. Real-world data also has the strength that its validity as a realistic usage scenario is hard to dispute. The disadvantage, however, is that real-world data is hard control and hard to randomize as may be required by some experimental designs. I.e. the data does not necessarily behave as required by the experiment. For instance, the ground truth about real-world data is often unobtainable, which is needed in some types of evaluation. The other drawback of using real-world data is that the generality of the obtained results can be disputed, if only a few different data sets are used. This implies that the results of the experiments may appeal to a more narrow audience.

With simulated data, the approach is to obtain the data from a simulated version of some real-world process. As such, this approach is very similar to real-world data in terms of pros and cons. However, the simulated data can be somewhat controlled, although most, if not all, real-world processes feature some degree of randomness/noise. On the other hand, the simulated data will never be more realistic than the model used for generating the data. Thus, in a sense, it becomes a poor man's version of real data. This means that a new source of criticism is opened: The quality and applicability of the simulation model.

Synthetic data is data which is generated without any specific real-world source in mind. As such, synthetic data can feature any desired shapes and structures. This provides the advantage of a very high degree of experimental control. For instance, this means that the experimental scenarios can be designed such that the ground truth perfect result is both known and, in most cases, achievable by the participants. Furthermore, it is still possible to relate synthetic scenarios to real-world scenarios by arguing that real-world scenarios can be decomposed into a number of simpler building blocks that resemble the synthetic scenarios. This implies that synthetic data has a potentially broad appeal - also in applied contexts. The main problem with synthetic data is the disconnect from the real-world. Even though randomness can be used in generating the data, the high

degree of control implies that no real-world condition will be exactly like the synthetic scenario.

In this thesis, the choice is to use synthetic data. There are multiple reasons for this choice:

1. The high degree of experimental control offered by synthetic data is attractive.
2. This thesis is not made with any specific application area in mind.
3. The notion that synthetic scenarios can be viewed as approximate parts of real scenarios seems to justify that the obtained results are likely to be generalizable to a broader context.

2.3 Outline of Experimental Procedures and Analyses

With the overall method in place, it becomes possible to outline the common traits of all the experiments carried out, and the subsequent analyses made. The archetypical experiment used in this thesis is a multi-factorial design, which features the following traits:

- The factors are *fully crossed*. Therefore, the design is a *complete factorial design*. I.e. all possible combinations of factor levels have been tested.
- The experiments are *balanced* and *replicated*. As such, the aim is to get an equal amount of responses for every experimental condition tested.
- The experiments are *within-subjects designs*, which means that each participant is exposed to all possible experimental conditions. This is done to be able to model and eliminate the influence of individual differences, implying that any statistically significant results found are not caused by the skill levels of individuals, but rather by the tested factors.
- The sequence of test conditions is *randomized* for each participant to level out any bias caused by learning during the experiment.
- Participants are asked to perform the same trial several times. This is a type of pseudo-replication, but this fact is taken into account in the data analysis.
- Participants do a few practice sessions at the beginning of each experiment to gain a minimum level of proficiency before logging any experimental data.
- The experiments are designed such that the average participant will spend approximately one hour doing the experiment, including breaks and practice sessions. This is done to avoid problems of exhaustion and tiredness on behalf of the participants. The average time for a single participant is estimated by making small-scale pilot tests with one or two participants before doing the full experiment.

In statistical terms, the best overall description of the experimental design would be a randomized complete block design (RCBD) [39], where the participants constitute the blocks of the experiment. Since the participants are randomly picked, they will be modelled as random effect blocks in the analysis. The factors of the experiment will be modelled as fixed effects. This implies that the statistical model used is going to be a mixed effects model. Hence, the primary analysis method will be type III ANOVA.

The use of ANOVA as preferred primary analysis tool implies that the risk of falsely rejecting null hypotheses[1] is not adversely affected by the number of experimental conditions. The significance level $\alpha = 0.05$ is used in all cases. I.e. the accepted risk of getting false positive results is 5%. The specific choice of factors, levels, and responses is based on a number of hypotheses stated prior to designing the experiments.

As a follow-up to the ANOVA tests, Tukey's honest significant difference test is used when ANOVA indicates that significant effects are present. This post-hoc analysis is used in order to identify exactly where the significant differences of factor levels are. In cases where the assumptions of ANOVA are violated beyond simple repair, the standard fallback strategy is to employ the non-parametric Friedman test instead. The preferred way of reporting significance is through p-values.

2.4 Methodology Summary

In summary, the overall scientific philosophy guiding the work of this thesis is the critical rationalism approach introduced by K. R. Popper. This implies that statistical methods is the preferred choice of analysis method. Consequently, this imposes some constraints on the measurements made in the experiments of the thesis. The main constraint is that the data must be of a quantitative nature. However, both objective and subjective measurements will be used to be able to evaluate the results from two different viewpoints.

The data gathered is preferably of ratio level, since this allows for the broadest range of analysis tools. However, in the case of subjective measurements on ordinal level scales, the use of ratio level analysis techniques will still be applied, based on recent results in statistical literature. Any other violations of the basic assumptions of the statistical methods used will be reported on a case-by-case basis.

Synthetic data is chosen to be the basis of any data visualizations. Although only being indirectly related to real-world applications, the amount of control offered by synthetic data was judged to outweigh the disadvantages. This choice does not imply that future work should not be based on real-world data, nor does it imply that studies on real-world data are in any way irrelevant or uninteresting.

[1] Called a type I error, or a false positive.

3 Box Geometry and Representations

3.1 Introduction

Rectangular 3-D boxes feature prominently as user-definable volumes of space in all of the three papers presented in this thesis. For this reason, this chapter presents some of the geometric aspects of 3-D boxes which forms the mathematical foundation of the designed box shaping techniques. Thus, this chapter deals with the geometry of boxes, and how to represent them in a format which is both useful for computational purposes, and for a user shaping a 3-D box.

The rectangular 3-D box[1], is one of the most common, primitive 3-D shapes used in modelling and construction in the real world. The reason for this may well be that rectangular boxes feature several nice properties in one simple shape:

1. Boxes are plesiohedra (space-filling polyhedra), meaning that it is possible to stack them without wasting space.

2. Boxes can be systematically stacked to produce larger rectangular boxes, making them convenient shapes for packaging and storage.

3. Boxes are convex, all angles are right, all opposite surfaces are parallel, making them highly symmetrical and simple to stack in an environment constrained by gravity.

4. Perhaps as a consequence of the three former properties, rectangular shapes are very often encountered in man-made objects, making them easy to relate to for most people. Well-known examples of rectangular boxes used in construction are floor tiles and bricks, see Figure 3.1.

In immersive, 3-dimensional virtual worlds, the rectangular box is not only useful for construction and storage. It also offers a way of letting a user define a simple, limited volume of space, e.g. for selection tasks. The potential of a rectangular box as a selection volume is best demonstrated in environments where multiple objects are to be selected at the same time. An analogy of this would

[1] Rectangular boxes are also known under several other names in literature such as right cuboid, regular/rectangular hexahedron, and rectangular parallelepiped.

Figure 3.1: Examples of box shapes used in real world construction. *(Left)* A wall made from clay bricks. *(Right)* Floor tiles. These two images demonstrate the practical stacking and tiling properties of rectangular boxes.

be the 2-D rubber band selection technique found in most desktop computing environments.

Defining and Constraining the Problem

There are 4 general steps to take into account when considering the creation and evaluation of 3-D box shaping techniques:

1. Understanding the geometry of a box and the data necessary to represent a box.
2. Creating box shaping techniques considering human aspects such as usability and perception.
3. Finding requirements for hardware and software.
4. Testing the techniques.

This chapter mainly deals with the first item in the list above, whereas the presented contributions of this thesis treats the remaining three items in various ways.

3.2 Basic Box Geometry

To fully understand the problem of defining a box in 3 dimensions, the first step is to look at the geometry of a box. Even though non-rectangular 3-D boxes exist, such as parallelepipeds and the even more general cuboid[2], the remainder of this document will refer to 3-D rectangular boxes, simply as boxes, unless the context indicates otherwise.

[2] Although various definitions of a cuboid exist, a cuboid in this chapter is taken to be any polyhedron bounded by six quadrilateral sides such that any of its 8 vertices coincide with exactly 3 of its faces. Some other sources take cuboid and a rectangular box to mean the same.

Box Definition

To define more precisely what is understood by an arbitrary box, it is useful to inspect the graph that can be produced from the edges and vertices of a box. The graph of any box shape must be isomorphic to this graph. The graph is shown in Figure 3.2.

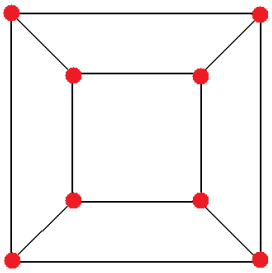

Figure 3.2: The box graph. The graph shows the vertices (red) and edges (black) of a box. The notable features of the graph are (1) it is planar, (2) it is of order 8, size 12, has 6 faces, (3) that all vertices are of degree 3. This graph is equally valid for rectangular as well as non-rectangular boxes. Indeed, one way of defining the most general form of box, the cuboid, would be any non-self intersecting and non-degenerate polyhedron with a graph isomorphic to the box graph.

Even though the graph in Figure 3.2 defines the overall layout of a rectangular box, it also defines any other box shape, i.e. any cuboid. Thus, box graph isomorphism is a necessary, but insufficient condition to fulfil.

To make the conditions sufficient, another constraint is needed that deals with the rectangular property of the boxes of interest in this case. This extra constraint is quite easy to define using the box graph: *Any two adjacent edges of the box graph must be orthogonal when its vertices are represented as points in $\mathbb{R}^3$, and all edges must be of non-zero length*. This condition ensures that the faces of the box remain rectangular and orthogonal, and that the box does degenerate into another shape such as a rectangle or line.

In brief, the two necessary and sufficient conditions to be fulfilled for an arbitrary rectangular box are:

1. The graph made from the edges and vertices of the polyhedron must be isomorphic to the box graph.

2. Any two adjacent edges of the box must be orthogonal to each other, when the vertices of the box graph are considered as points in $\mathbb{R}^3$, and all edges must be of non-zero length.

Combining the two conditions produces the familiar rectangular box shape shown in in Figure 3.3.

With the definition of a box in place, it is useful to define some terminology related to boxes.

Figure 3.3: The same rectangular box rendered in three different ways. *(Left)* Rendered using the simple Blinn-Phong reflection model *(Middle)* A transparent version of the box, where both the external and internal faces are visible. *(Right)* Rendered using 2-sided wireframe mode. The wireframe version reveals that the box is, indeed, isomorphic to the box graph.

A vertex of a box graph when taken to be a point in $\mathbb{R}^3$ is also called a *corner* of the box. There are three ways of connecting the corners of a box:

1. A line segment connecting two corners with a box graph distance of 1 is called a *box edge*. The edges of the box graph correspond to box edges. Thus, any box has exactly 12 box edges. Each box edge has 3 parallel counterparts on the box.

2. A line segment connecting two corners with a box graph distance of 2 is called a *face diagonal*. Two diagonals can be made for each face of a box, yielding a total of 12 different face diagonals for any box. Each face diagonal has 1 parallel counterpart on the opposite face of the box.

3. A line segment connecting two corners with a box graph distance of 3 will be called a *box diagonal*[3]. A box has 8 corners, each being the endpoint of 1 box diagonal. This yields a total of 4 different box diagonals in a box. None of the box diagonals are parallel.

The three line segments are illustrated in Figure 3.4.

Box Dimensions and Measurements

The three primary dimensions of a box are measured along three orthogonal box edges. These measurements are often referred to as the width, the height, and the depth of the box. All of these lengths must be positive to avoid getting into conflict with the conditions defining a box.

From these three primary dimensions, all other standard measurements of a box can be derived. Assuming that the three dimensions are represented by three real numbers, a, b, and c, the other measurements related to the box can be derived.

[3]Box diagonals are also sometimes referred to as space diagonals in literature.

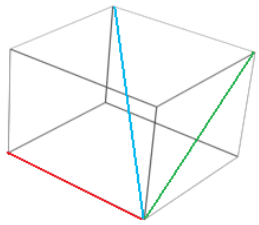

Figure 3.4: A box with examples of a box edge (red), a face diagonal (green), and a box diagonal (blue) illustrated.

Measurements Related to Individual Faces

Any measurement related to a face is related to the dimensions of that face. Since each face is defined by two dimensions, any formula for a face measurement must also involve two of the primary box dimensions. A box has, at most, three different types of faces, since opposite faces share the same dimensions. Thus, any measurement relating to an individual faces comes in three different varieties depending on the dimensions relating to that face.

The face areas, S,can be computed as follows:

$$S_{a,b} = ab \quad S_{a,c} = ac \quad S_{b,c} = bc \tag{3.1}$$

The face diagonal lengths, l, can be computed using Pythogoras' formula:

$$l_{a,b} = \sqrt{a^2 + b^2} \quad l_{a,c} = \sqrt{a^2 + c^2} \quad l_{b,c} = \sqrt{b^2 + c^2} \tag{3.2}$$

Each face also has a perimeter, p, which is computed as follows:

$$p_{a,b} = 2(a + b) \quad p_{a,c} = 2(a + c) \quad p_{b,c} = 2(b + c) \tag{3.3}$$

Measurements Related to an Entire Box

The first measurement relating to the whole box is the total surface area of the box, S_{box}. This quantity can be computed as two times the sum of the three face areas, since each face is represented twice:

$$S_{\text{box}} = 2(ab + ac + bc) \tag{3.4}$$

The next box measurement is the volume of the box, V_{box}, which is a simple multiplication of a face area by the dimension missing from that face:

$$V_{\text{box}} = abc \tag{3.5}$$

The final box measurement is the length of the box diagonal, $l_{a,b,c}$. Although none of the box diagonals are parallel, they are of the same length as long as long as the box is rectangular. The formula for the length can be derived using Pythagoras' formula:

$$l_{a,b,c} = \sqrt{\sqrt{a^2 + b^2}^2 + c^2} = \sqrt{a^2 + b^2 + c^2} \tag{3.6}$$

Symmetries of a Box

The symmetries of a box are important to understand as a prerequisite to examining the effect of applying various transformations to the box.

From a graph theoretical point of view, the vertices of a box are highly similar. The fact that every vertex of the box graph has degree 3 means that it is impossible to distinguish each vertex, unless some edge or vertex colouring scheme is employed to distinguish them.

One meaningful way of augmenting the box graph with some extra information about the geometry of the box is to colour the box graph's edges according to the edge lengths. The orthogonality condition implies that any box has at most 3 different edge lengths. Thus, an edge colouring encoding the dimensions of the box needs only three different colours. Doing so produces the graph shown in Figure 3.5.

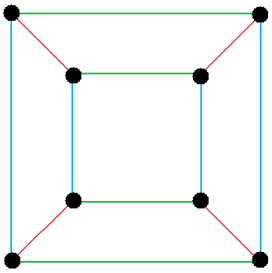

Figure 3.5: The box graph augmented with coloured edges. The edge colours encode the lengths of the edges. This colouring shows that all vertices are connected to a red, a green, and a blue edge. Thus, this colouring has not made the vertices any less similar than they were in the simple box graph.

As Figure 3.5 shows, all vertices in the edge coloured box graph are still similar: Each vertex is connected to a red, a green, and a blue edge. This fact combined with the condition that adjacent edges are orthogonal, has two implications. First of all, it implies that the same results can be reached no matter which single vertex is picked from the box graph. Secondly, it means that boxes feature several geometrical symmetries.

First of all, a box has 3 planar mirror symmetries. Mirroring a box through a plane parallel to its sides, through the centre of the box, leaves the box unchanged. This is illustrated in Figure 3.6.

The fact that at least one mirror symmetry exists for boxes, implies that a box is an achiral[4] shape. I.e. mirroring a box through *any* plane can be reversed by

[4]A chiral shape is one whose mirror image cannot be mapped to the original figure through

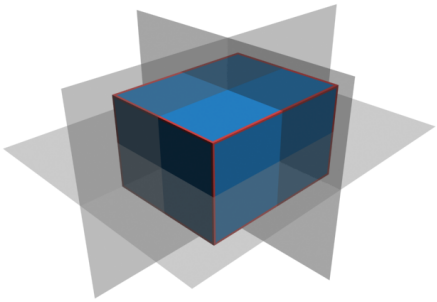

Figure 3.6: The three mirror symmetry planes of a rectangular box. Each plane bisects the box and is parallel to two of the box' faces.

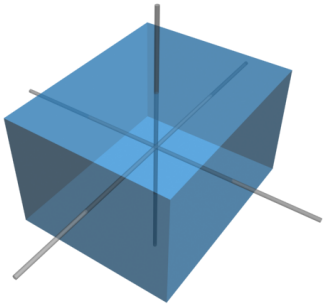

Figure 3.7: The three axes of rotational symmetry of a box.

some combination of rotation and translation transformations, or conversely that any mirroring can be expressed as a combination of rotation/translation instead. A box can also be mirrored through its centre point without changing the box.

The second kind of symmetry featured by a box is rotational symmetry. Rotating a box by any multiple of $180°$ about any axis perpendicular to the faces of the box through the box' centre, does not change the box. Figure 3.7 shows these axes.

For cubes, the amount of symmetry increases, since $90°$ and $270°$ rotations about the symmetry axes are also allowed. A cube has a total of 48 different symmetries, consisting of 24 rotations and 24 reflections. These symmetrical properties are of interest in the following two sections.

Boxes and Coordinate Systems

The orthogonality condition given in section 2.1 means that a Cartesian 3-D coor-

a combination of translation and rotation. Human hands are prime examples of chiral shapes (you cannot fit a left glove on your right hand merely by translating/rotating the glove). Being achiral/amphichiral means the opposite of being chiral.

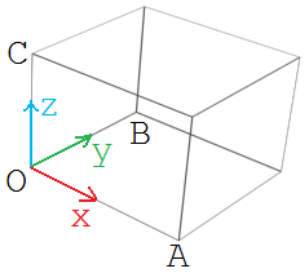

Figure 3.8: A box coordinate system constructed from 4 corners, an origin corner, O, and its three neighbours, A, B, and C. The direction of the three axes are calculated by normalizing the three point differences $A - O$, $B - O$, and $C - O$. The origin may also conveniently be placed at the centre of the box in some cases.

dinate system can easily be defined by selecting an arbitrary corner of the box as origin[5], O, and using its 3 neighbouring corners, (A,B,C), to construct 3 orthogonal, normalized vectors as axes. The justification for allowing an arbitrary choice of origin corner was given in the previous section. Such a coordinate system will be referred to as a box coordinate system. Figure 3.8 illustrates a box coordinate system.

A box of dimensions $a \times b \times c$ represented in its own box coordinate system always has its 8 corners located at positions $[0,0,0]^T$, $[a,0,0]^T$, $[0,b,0]^T$, $[0,0,c]^T$, $[a,b,0]^T$, $[a,0,c]^T$, $[0,b,c]^T$, $[a,b,c]^T$, which makes the box coordinate system very convenient to use for calculations involving that particular box.

Since there are 8 corners on a box, 8 different origins can be chosen for a box coordinate system. The 3 axes along the edges of the box can also be freely permuted, i.e. any of the three vectors can be the box' x-axis, etc. As there are $3 \times 2 \times 1 = 6$ permutations of axes for each origin, this yields $6 \times 8 = 48$ possible box coordinate systems with corner origins for a single box. If the coordinate systems are constrained to being either right-handed or left-handed, this halves the number of possibilities. Incidentally, 48 is also the total number of symmetries possible for a cube.

Because of box symmetries, any axis in one box coordinate system always has parallel counterpart in each of the other possible box coordinate systems. This implies that any of the 48 box coordinate systems are equally convenient for describing the positions of the corners of the box.

The polyhedron defined by points O, A, B, and C is called a *trirectangular tetrahedron*. Trirectangular because the edges meeting at point O, also called the

[5]Since a change of origin amounts to a vector subtraction for all affected points, the origin of a box coordinate system is not as important as the axes of the coordinate system being parallel to the edges of the box. For this reason, the origin of the coordinate system can be placed anywhere in relation to the box, not just at its vertices, without large practical complications. E.g. an origin at the centre of a box may be just as convenient as an origin at a vertex of a box, in some cases.

apex, are orthogonal; tetrahedron because it is bounded in space by four triangular faces. The triangle formed by points A, B, C, $\triangle ABC$, is called the base of such a tetrahedron. This terminology and the properties associated with trirectangular tetrahedra become useful later on in section 3.3.

The corners of a box may also be described in some common frame of reference, called a world coordinate system. This is especially useful if working with more than one virtual box.

If the box and world coordinate systems have parallel axes, the box is said to be *axis-aligned*. Axis-aligned boxes are very simple to test for overlaps, which is why they are extensively used in real-time collision detection systems [14]. Axis-aligned bounding boxes used in collision detection are often referred to as AABB's. A box which is not axis-aligned is sometimes said to be oriented.

Degrees-of-Freedom and Transformations of a Box

Even though the sufficient conditions for defining a box are given in section 2.1, they reveal little about the degrees-of-freedom (DoF) of the box, i.e. the number of independently controllable parameters governing a box. The number of DoF is of interest, because it tells the minimum number of independent parameters that must be defined, e.g. by a user, before a box can be unambiguously constructed.

Although the box is fully defined if the positions of all 8 corners are given, the constraints implied in the box definition mean that the information contained in the 24 parameters needed to define 8 points in 3-D space are, to a large extent, redundant.

One simple way of deriving the correct number of degrees of freedom is to evaluate the independent transformations that can be allowed without violating any of the defining conditions for a rectangular box.

First of all, any rigid transformations, which by definition do not alter any internal angles or dimensions of an object, are permissible. This means that at least translation and rotation must be included. Furthermore, since we desire a box of arbitrary dimensions, scaling, although non-rigid, also needs to be included.

Shear and other transformations that do not preserve the internal angles of a shape would violate the orthogonality condition for rectangular boxes, so this class of transformations can be immediately disregarded, as it would transform the rectangular box into a more general parallelepiped.

Since the investigation of box symmetry revealed boxes to be achiral, there is no need to add reflections to the list of transformations to be investigated. This is due to the fact that for achiral shapes, reflection through an arbitrary plane is equal to some combination of rotation and translation. Thus, reflection as a box transformation features no new degrees of freedom that cannot be expressed through other transformations.

Translation as DoF

Translation is the transformation that takes care of our requirement of being able to place the box at an arbitrary position.

The transformation is carried out by adding the same vector to all the corners of the box. Translation of a single point, $\boldsymbol{p}$, by a vector, $\boldsymbol{v} = [x, y, z]^T$, can be expressed mathematically as follows:

$$T_{\boldsymbol{v}}(\boldsymbol{p}) = \boldsymbol{p} + \boldsymbol{v} \tag{3.7}$$

If the point is defined using homogeneous coordinates, translation can also be carried out as a matrix multiplication, which is most often the case in computer graphics. If $\boldsymbol{p}$ is of the form $[x_p, y_p, z_p, 1]^T$, and $\boldsymbol{T_v}$ is a 4×4 translation matrix, then:

$$T_{\boldsymbol{v}}(\boldsymbol{p}) = \boldsymbol{T_v p} \tag{3.8}$$

Since the translation vector in 3-D has three independent components, one for translation along each axis, translation in 3-D has 3 DoF.

Rotation as DoF

Whereas translation takes care of the arbitrary box positioning requirement, rotation takes care of the arbitrary orientation of the box to be created.

In 3-D, a rotation can be represented in several ways. Regardless of representation, all rotations occur around an axis through the origin. If this is not the desired outcome, then the rotation transformation must be sandwiched between two equal but opposite translations.

Two very common rotation approaches are Euler angles and angle-axis.

In an Euler angle approach, an arbitrary rotation is composed of 3 independent rotations about 3 fixed, orthogonal axes in a fixed sequence. Since the axes and the sequence is fixed, only the 3 rotation angles need to be defined. This is also the minimal number of parameters to represent an arbitrary 3-D rotation. Unfortunately, Euler angles suffer from a problem known as gimbal lock, where two of the rotation angles affect the same axis, effectively reducing the number of independent angles from 3 to 2. This is a consequence of composing the total rotation from 3 separate rotations in a fixed sequence.

Euler angle rotation is often implemented using three rotation matrices, one for each angle, (α, β, γ), around axes ($\boldsymbol{v}_1$, $\boldsymbol{v}_2$, $\boldsymbol{v}_3$), respectively[6]. The matrices can be 3×3 or 4×4 depending on the format of the points to be rotated:

$$R_{\alpha,\beta,\gamma}(\boldsymbol{p}) = \boldsymbol{R}_\gamma \boldsymbol{R}_\beta \boldsymbol{R}_\alpha \boldsymbol{p} \tag{3.9}$$

Angle-axis rotation is equivalent to Euler angle rotation, which means that it still has 3 DoF. However, it is gimbal lock free. The rotation takes place around an arbitrary axis, $\boldsymbol{r}$, by an arbitrary angle, θ . Even though a rotation axis has 3 parameters, and the angle is 1 parameter, the angle-axis format can still be encoded as 3 numbers, because rotation is independent of the length of the rotation axis. Such an encoding of angle and axis is called a rotation vector. For practical

[6]Axes $\boldsymbol{v}_1$ and $\boldsymbol{v}_3$ of an Euler angle rotation can be the same coordinate axis, since an individual step of a composite rotation also affects all subsequent rotation axes of the composition.

reasons, angle-axis rotation is usually implemented in one of two other formats instead.

As a rotation quaternion [29], $\boldsymbol{q}_{\boldsymbol{r},\theta}$:

$$R_{\boldsymbol{r},\theta}(\boldsymbol{p}) = \boldsymbol{q}_{\boldsymbol{r},\theta/2}\boldsymbol{p}{\boldsymbol{q}_{\boldsymbol{r},\theta/2}}^{-1} \tag{3.10}$$

As an angle-axis rotation matrix, $\boldsymbol{R}_{\boldsymbol{r},\theta}$:

$$R_{\boldsymbol{r},\theta} = \boldsymbol{R}_{\boldsymbol{r},\theta}\boldsymbol{p} \tag{3.11}$$

The conclusion is that regardless of approach to rotation, the requirement of arbitrary box orientation adds 3 DoF to the total.

Scaling as DoF

Scaling is necessary to fulfil the requirement of an arbitrarily sized box. There are two overall types of scaling, *uniform* and *non-uniform* scaling. To avoid degenerate cases, it is clear that a scaling factor of 0 cannot be permitted, since this would degenerate the geometry of the box.

For uniform scaling, all axes of an object are scaled (multiplied) by an equal non-zero amount. This implies that uniform scaling preserves some of the same properties as a rigid transformation: Internal angles are preserved as well as all size ratios. Straight lines also remain straight after transformation. However, unlike rigid transformations, internal distances are not preserved. Since uniform scaling is a multiplication of all corners by a single scale factor, s, it is a transformation with 1 degree of freedom.

$$S_s(\boldsymbol{p}) = s\boldsymbol{p}, \quad s \neq 0 \tag{3.12}$$

Uniform scaling can also be expressed as a matrix multiplication using an identity matrix along with the scaling factor:

$$S_s(\boldsymbol{p}) = (s\boldsymbol{I})\boldsymbol{p}, \quad s \neq 0 \tag{3.13}$$

With non-uniform scaling, each coordinate axis has a separate scale factor applied. In 3-D this means three independent scale factors, yielding 3 DoF. If all 3 scale factors are equal, the non-uniform scaling reduces to a uniform scaling. Thus uniform scaling is a special case of non-uniform scaling.

For most shapes, non-uniform scaling changes both the internal angles, the internal distances, and the ratio of internal distances. The only preserved properties are straight lines remaining straight, and distance ratios along a line remaining the same. The possible change of internal angles indicates that non-uniform scaling can violate the orthogonality condition for boxes, which would force non-uniform scaling to be disallowed. This conclusion, however, depends on the coordinate system in which the non-uniform scaling takes place. An example of non-uniform scaling going wrong is shown in Figure 3.9.

The orthogonality condition implies that as long as the non-uniform scaling takes place in a box-aligned coordinate system, no internal angles will change.

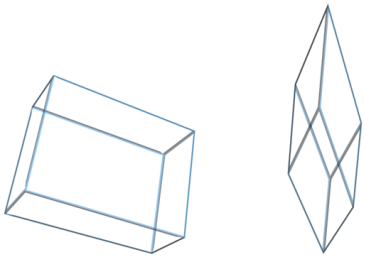

Figure 3.9: Non-uniform scaling of the left rectangular box has caused it to transform into the right, non-rectangular box – a parallelepiped. This violates the orthogonality condition for rectangular boxes.

This fact can be readily seen from the definition of orthogonality of two non-zero vectors, e.g. two edge vectors of a box:

$$\forall \boldsymbol{a}, \boldsymbol{b} \neq \overline{0} : a \perp b \Leftrightarrow \boldsymbol{a}^T \boldsymbol{b} = \|\boldsymbol{a}\| \|\boldsymbol{b}\| \cos 90^\circ = 0 \tag{3.14}$$

The observation that the lengths of the orthogonal vectors, $\boldsymbol{a}$ and $\boldsymbol{b}$, in the above equation do not affect the value of their inner product means that they can be independently scaled by any non-zero factor without violating the orthogonality condition.

Non-uniform scaling is expressed mathematically as a multiplication of a diagonal matrix, $\boldsymbol{S}_{s_x,s_y,s_z}$ onto the points/vectors to be scaled:

$$S_{s_x,s_y,s_z}(\boldsymbol{p}) = \boldsymbol{S}_{s_x,s_y,s_z}\boldsymbol{p}, \quad s_x, s_y, s_z \neq 0 \tag{3.15}$$

The conclusion on non-uniform scaling is therefore that it adds another 3 DoF, and that it must be applied before any rotation of the box out of its own coordinate system takes place.

Handedness

The results of rotation and translation transformations can only be determined unambiguously, if the handedness of the coordinate system that they take place in has been determined. Two examples of this ambiguity are:

1. Given a translation vector of $[1, 0, 0]^T$, it is necessary to know if it physically corresponds to translating 1 unit to the right, or 1 unit to the left relative to the current position - even if you know the direction of the other two axes.

2. If given a rotation axis and angle, you must know if the rotation angle is to be applied clockwise or counter-clockwise around the axis.

Thus, a handedness ambiguity will always exist in any box representation where no convention of handedness has been settled a priori. For this reason, the

handedness issue must be settled first. In the rest of this report, it is therefore implicitly assumed that the handedness of the space is known a priori. We assume that all coordinate systems are right-handed, unless otherwise stated.

Box DoF and Transformation Summmary

After examining the possible transformations that we can allow a box to undergo, it has been determined that a box of arbitrary position, orientation, and size is subject to 9 degrees-of-freedom. This means that 9 independent parameters must be provided in any minimal representation of a box. If anything less than 9 independent parameters are given, then the representation will be ambiguous to some extent.

If the box is axis-aligned, the implication is that the 3 rotational degrees of freedom have been fixed. This means that an axis-aligned box only has 6 degrees of freedom, making it somewhat simpler to define than a freely oriented box. These 6 parameters could e.g. be given as the end points of a box diagonal of the desired axis-aligned box.

The investigation also showed that all three transformation types can be implemented as matrix multiplications, which is practical since the transformations can then be expressed as a single, combined matrix.

Furthermore, investigation of non-uniform scaling revealed that it must be carried out in the box' own coordinate system before any rotation is made. In order to make the box rotate on an axis in its own coordinate system, the box should also be rotated before translating it.

The recommended sequence for applying the transformations to a point in a box coordinate system, $\boldsymbol{p}_{\text{box,local}}$, is therefore scale then rotate then translate, which looks as follows:

$$\boldsymbol{p}_{\text{box,world}} = \boldsymbol{T}_{x,y,z}\boldsymbol{R}_{\boldsymbol{r},\theta}\boldsymbol{S}_{s_x,s_y,s_z}\boldsymbol{p}_{\text{box,local}}, \quad s_x, s_y, s_z \neq 0 \tag{3.16}$$

3.3 Box Representations

Having established various definitions and terminology in the previous sections, the next step is to look at practical ways of representing and constructing a box in terms of different parameters.

Even though it was established in section 3.2 that the 8 corners of a box contain redundant information, it is necessary for rendering purposes to know all 8 corners explicitly. For this reason, each box representation examined should both contain the data of the representation and an algorithm for computing the explicit data for all 8 corners.

Before diving into specific representations, it is useful to realize what the design criteria that we seek to fulfil are. The main design criteria for a useful box representation are:

1. Compactness. A good representation minimizes the amount of data stored/ needed, leading to smaller memory consumption by an implementation us-

ing the representation. Good compactness also implies less work on behalf of a user defining the box.

2. Unambiguity. A good representation only allows one box to be constructed from its data.

3. Practicality. A good representation is designed with the intended use of the box in mind. If the box representation fits the typical use of the box, the computational load of using it can be reduced.

It is expected that in most cases a box representation is chosen based on some compromise between these three. For instance, compactness may very often conflict with the other two criteria.

3.4 3-Corner Representations of a Box

The first family of representations examined are representations that contain the positions of three corners of a box. Such representations have not been previously considered in literature. The reasons for choosing this approach are twofold:

1. Three positions offer 9 parameters/numbers, which was the bare minimum according to the DoF analysis in section 3.2. However, 9 parameters are only enough, if they can be independently controlled.

2. By investigating representations at or near the minimum requirements, many observations can be made that can serve as inspiration for other representations and construction techniques for boxes.

This leads to two steps of analysis that must be made:

1. How many ways can three corners be picked from a box?

2. For each of the ways found in step 1:

   a) Are the 9 parameters all independently controllable?

   b) How to find the remaining 5 corners of the box?

Picking 3 Corners from a Box

For reasons of symmetry and self-similarity, the first of the three corners can be picked arbitrarily, meaning that its three coordinates are all independent. This first corner will be referred to as A.

The second corner picked, B, can either be adjacent or non-adjacent to A in the box graph. Since two points alone do not define any of the internal angles of the box, B can also be chosen arbitrarily as long as it is not identical to A. Thus all 6 parameters of A and B are independent.

The third and final corner, C, has several possibilities in relation to A and B:

1. If A and B are adjacent, then line AB forms a *box edge*:

a) C is adjacent to either A or B, but not both. $\triangle ABC$ is bounded by *two box edges and a face diagonal*.

b) C is non-adjacent to both A and B. $\triangle ABC$ is then bounded by *a box edge, a face diagonal, and a box diagonal*.

2. If A and B are non-adjacent and line AB forms a *face diagonal*:

a) C is adjacent to both A and B. $\triangle ABC$ is bounded by *two box edges and a face diagonal*.

b) C is non-adjacent to both A and B. $\triangle ABC$ is bounded by *three face diagonals*.

3. If A and B are non-adjacent and line AB forms a *box diagonal*:

a) C is adjacent to either A or B, but not both. $\triangle ABC$ is bounded by *a box edge, a face diagonal, and a box diagonal*.

This adds up to a total of five cases (1.a, 1.b, 2.a, 2.b, and 3.a). Each of these cases are illustrated in Figure 3.10.

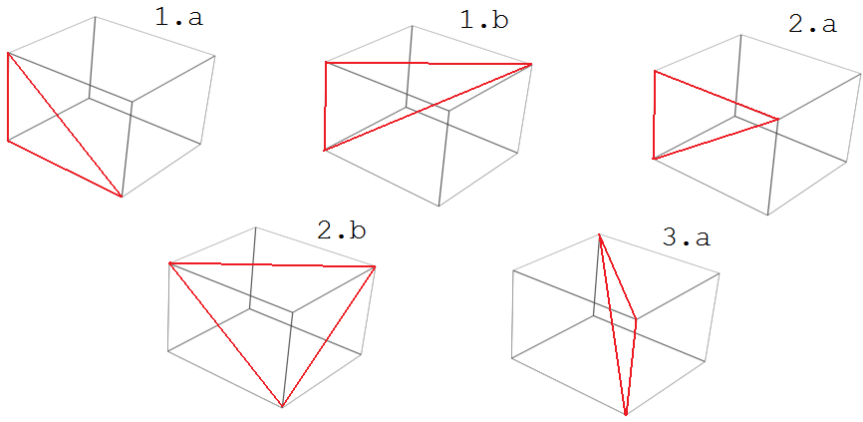

Figure 3.10: The five possible pickings of 3 corners on a box. A brief inspection of the red triangles reveals that 1.a and 2.a are essentially the same triangle in relation to the box vertices, which means that they can be merged. 1.b and 3.a can also be merged for the same reason. This leaves only 1.a, 1.b and 2.b to examine.

The five 3-corner pickings lined out above can be reduced by grouping them by the way the triangle edges relate to the box graph. The justification of grouping the triangles in this way is the similarity of vertices/corners, the symmetries of boxes, and the fact that corners A, B, and C can be picked in any order without affecting the resulting triangle. This leaves only three cases to consider in this section:

I A triangle formed by three face diagonals.

II A triangle formed by a box edge, a face diagonal, and a box diagonal.

III A triangle formed by two box edges and a face diagonal.

In any of the three cases, the information needed can either be given directly as 3 corner positions (A, B, and C), or as one corner, A, and 2 direction vectors, ($\boldsymbol{u}, \boldsymbol{v}$), pointing to the remaining 2 corners of the triangle. These two representations are equivalent to each other, since given A, B, and C, the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ can be determined unambiguously (and vice versa):

$$\left\{ \begin{array}{l} B = A + \boldsymbol{u} \\ C = A + \boldsymbol{v} \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{l} \boldsymbol{u} = B - A \\ \boldsymbol{v} = C - A \end{array} \right\} \tag{3.17}$$

The determination of the completed box from either representation can therefore be made in the same way, which means that the choice between the two can be made freely with no consequence to the results.

Case I: 3 Face Diagonals

In this case the three corners have been picked such that they are all non-adjacent to each other in the box graph. Such a representation is immediately attractive by virtue of the similarity of the relation between the three corners and the box graph. I.e. the corners all connect through face diagonals, and therefore none of corners play a special role compared to the other two.

The problem to be solved is to find the remaining 5 corners. One simple approach to solving this is a two step process:

1. Find a 4^{th} corner, D, which is non-coplanar with $\triangle ABC$. In this case, none of the other corners are coplanar with $\triangle ABC$, which makes them all equally usable in this context.

2. Use simple vector arithmetic to find the remaining 4 corners.

The 4^{th} corner, D, that we aim to find can, in principle, be any of the missing corners of the box. However, to approach the solution to the first step in a systematic way, the corner is chosen such that A, B, C, and D forms a trirectangular tetrahedron. In other words, D can be viewed as the origin of a box coordinate system with axes pointing towards A, B, and C, respectively. This is shown in Figure 3.11.

Constraints on the Triangle

The first task is to determine the constraints on $\triangle ABC$ which ensure that the box will be rectangular. A simple way of doing this is to represent the points A, B, and C in the box coordinate system where D is origin. This can be done without loss of generality. The coordinates of A, B, and C in this frame are $A = [a, 0, 0]^T, B = [0, b, 0]^T, and C = [0, 0, c]^T$. These coordinates can then be used to compute the edge lengths $|AB|$, $|BC|$, and $|AC|$. Doing so yields:

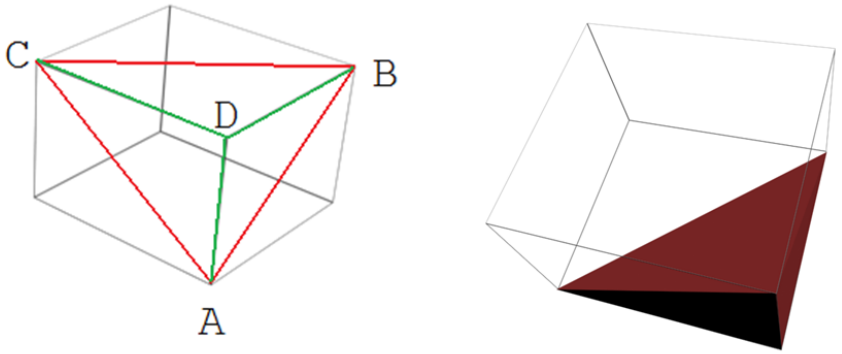

Figure 3.11: *(Left)* The three points A, B, and C have been chosen such that $\triangle ABC$ has edges along three face diagonals of the box. The unknown point, D, which must be found, is placed relative to $\triangle ABC$ such that D is the apex of a trirectangular tetrahedron with $\triangle ABC$ as base. An alternative formulation of this is that D is the origin of a box coordinate system with axes pointing towards A, B, and C, respectively. *(Right)* A rendering of a trirectangular tetrahedron inside a box. The volume of the tetrahedron is $1/6$ of that of the entire box.

$$\begin{aligned} |AB|^2 &= (A-B)^T(A-B) = a^2 + b^2 \\ |BC|^2 &= (B-C)^T(B-C) = b^2 + c^2 \\ |AC|^2 &= (A-C)^T(A-C) = a^2 + c^2 \end{aligned} \tag{3.18}$$

It is worth noting that for a non-degenerate box, both a, b, and c must be positive numbers. If these squared lengths are substituted into Pythagoras' formula, the result is that the triangle cannot be right. The correct relations between the edges turn out to be:

$$\begin{aligned} |AB|^2 + |BC|^2 &> |AC|^2 \\ |AB|^2 + |AC|^2 &> |BC|^2 \\ |BC|^2 + |AC|^2 &> |AB|^2 \end{aligned} \tag{3.19}$$

These relations between the edge lengths correspond to the definition of an acute triangle, which means that $\triangle ABC$ must be acute if it is to fit inside a rectangular box. The implications of this are:

1. If a set of three points are given to be used as non-adjacent corners of a box, the first sanity check before proceeding with construction should be to check if the above relations hold for those three corners. Otherwise, construction of a rectangular box is impossible.

2. If two points, A and B, have been given, then the location of C must be restricted to the region in space that forms acute triangles with A and B, if this representation is to be valid.

3. If at least one solution for the apex, D, exists, given that $\triangle ABC$ is acute, then any acute triangle is the base of some trirectangular tetrahedron.

The fact that $\triangle ABC$ must be acute means that the position of C is not completely independent of A and B. The next interesting thing to determine is the region in 3-D space where C is allowed to be, once A and B are given. One easy way of finding this region is to see where $\triangle ABC$ would be right, since right triangles form the borderline case between acute and obtuse triangles.

If a line segment is constructed between A and B, then this line segment can serve as the diameter of a sphere placed on the centre point of the line segment, i.e. at position $(A + B)/2$. Any point, C, chosen on the surface of this sphere will form right triangles with A and B, except if C is identical to A or B. The justification of this is both stated in Thales' theorem and in the median theorem of right triangles. If C is chosen from the aforementioned sphere, then the line segment AB is the hypotenuse of the right $\triangle ABC$. This is illustrated in Figure 3.12.

Furthermore, if C is chosen from a plane through either A or B, perpendicular to AB, then $\triangle ABC$ must also be right. In that case, AB becomes a cathetus in $\triangle ABC$. The two planes will be parallel tangent planes to the sphere mentioned above. Once again the exception is if C is identical to A or B. This is also illustrated in Figure 3.12.

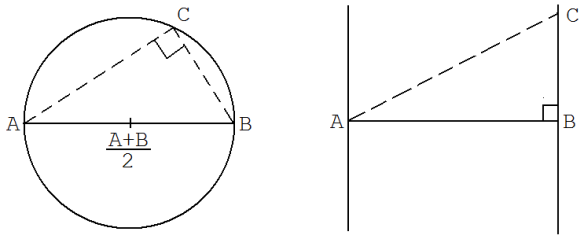

Figure 3.12: *(Left)* A 2-D outline of the sphere defined by the line segment AB. Any point, C, picked on this sphere forms a right triangle together with A and B. *(Right)* A 2-D outline of the two planes perpendicular to AB, through either A or B. Any point, C, arbitrarily chosen from either of these two planes also forms right triangles with A and B. The only exception is that C cannot be at the same position as A or B, since this degenerates the triangle into a line.

The sphere and two planes split the 3-D space into a number of regions where triangles ABC are either acute, right, obtuse, or degenerate. It has just been established that a point C chosen from either the circle or one of the two planes of Figure 3.12 produces *a right or a degenerate triangle*.

If C is located *inside the sphere* the triangle becomes *obtuse*, the extreme case being that C is located on the AB line segment, where the obtuse triangle degenerates into a line.

If C is located *outside the slab*[7] formed by the two planes, then $\triangle ABC$ is also

[7] A slab is defined as the space between two non-identical, parallel planes. Thus, a slab is fully defined by knowing the slab normal vector, the distance from the origin to the first plane, and the

obtuse. That is, unless C lies on the continuation of line segment AB, in which case $\triangle ABC$ degenerates into a line.

If C is located *outside the sphere but inside the slab*, then $\triangle ABC$ is *acute*. Thus, the region where a valid C for this representation can be chosen from is the set difference of a slab and a sphere.

All of the mentioned regions are illustrated in Figure 3.13.

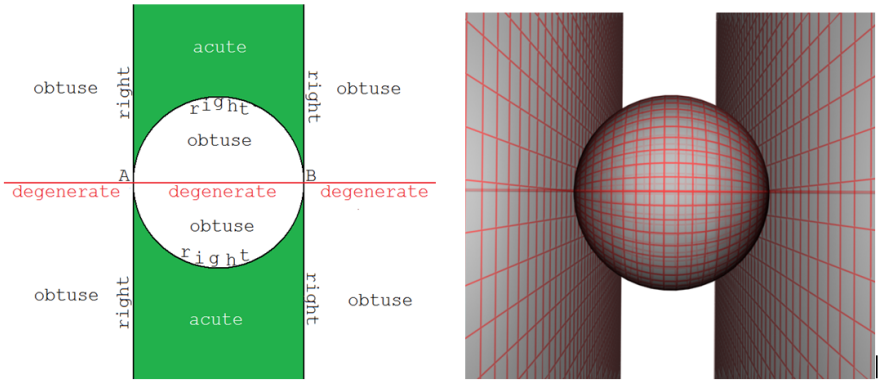

Figure 3.13: *(Left)* A 2-D view of the regions around points A and B, where picking a third point, C, forms different types of triangles, $\triangle ABC$. If the point is picked from one of the white regions, $\triangle ABC$ becomes obtuse. If C is picked from the green regions, $\triangle ABC$ becomes acute. If picked from the black boundaries (sphere or slab surface), $\triangle ABC$ becomes right. Finally, a point picked from the red line through A and B forms a degenerate triangle, i.e. a line. *(Right)* A 3-D view of the same regions. The points A and B are at the left and right extremes of the sphere, where it touches the two parallel planes. If C is picked from anywhere between the two planes, but outside the sphere, $\triangle ABC$ becomes acute. For practical reasons, the two planes in the rendering do not stretch out infinitely as they should. The thick, central, horizontal line is the triangle degeneration region.

Determining the 4^{th} Corner

Having established that all valid relationships between points A, B, and C are acute triangles, and where C must be located in relation to A and B, the next step is to determine the possible locations of the point D, which, together with A, B, and C, enables easy determination of all remaining corners (see Figure 3.11).

The constraints governing where D can be located in 3-D space is that the line segments DA, DB, and DC must all be perpendicular. This translates into the following system of three equations in three unknowns, the unknowns being the

distance between the two planes. This gives a total of 4 degrees of freedom since the length of the normal is of no consequence to the slab.

three coordinates of $D = [d_1, d_2, d_3]^T$. The basis of these equations is that the inner products of two non-zero vectors is 0 if and only if they are perpendicular:

$$\left\{ \begin{array}{c} (D-A)^T(D-B)=0 \\ (D-B)^T(D-C)=0 \\ (D-A)^T(D-C)=0 \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{c} D^TD-(A+B)^TD+A^TB=0 \\ D^TD-(B+C)^TD+B^TC=0 \\ D^TD-(A+C)^TD+A^TC=0 \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{c} D^TD-(A+B)^TD+A^TB=0 \\ (A-C)^TD+B^T(C-A)=0 \\ (B-C)^TD+A^T(C-B)=0 \end{array} \right\} \tag{3.20}$$

The steps taken above is an expansion of the inner products followed by a subtraction of the first equation from the other two equations. The fact that D^TD still occurs in the first equation means that this equation has quadratic terms for all of the unknowns. This has two implications:

1. The system as a whole is non-linear, so it cannot be solved by Gauss elimination[8], or by other techniques relating to linear systems.
2. The quadratic terms imply that the system can have up to 2 real-valued solutions for each coordinate of D. This means that two different D's, D_1 and D_2, are expected.

The reductions made to the system have transformed it from a system of 3 quadratic equations in 3 unknowns into 1 quadratic + 2 linear equations in 3 unknowns. The proposed solution method is as follows:

1. Solve the two linear equations using Gauss elimination. This leaves two expressions where d_1 and d_2 have both been expressed as linear functions of d_3.
2. Substitute d_1 and d_2 into the quadratic and solve for d_3. This produces the valid solution(s) for d_3, if any exists.
3. Substitute the solution(s) for d_3 back into the linear system to find the corresponding solution(s) for d_1 and d_2.

Performing Gauss elimination and back substitution on the linear system yields a result of the following form, where k, l, m, and n represent values that can be computed solely from the coordinates of the known corners A, B, and C:

$$\left[\begin{array}{ccc} 1 & 0 & k \\ 0 & 1 & m \end{array} \right] \left[\begin{array}{c} d_1 \\ d_2 \\ d_3 \end{array} \right] = \left[\begin{array}{c} l \\ n \end{array} \right] \Leftrightarrow \left[\begin{array}{c} d_1 \\ d_2 \\ d_3 \end{array} \right] = \left[\begin{array}{c} l \\ n \\ 0 \end{array} \right] + d_3 \left[\begin{array}{c} -k \\ -m \\ 1 \end{array} \right] \tag{3.21}$$

The second formulation of Equation 3.21 reveals that the solutions to the linear part of the system forms a line in space. The mathematical expressions

[8] At least not by Gauss elimination alone, but Gauss elimination is still a viable step of the solution procedure.

of k, l, m, and n are given below, where $A = [a_x, a_y, a_z]^T$, $B = [b_x, b_y, b_z]^T$, $C = [c_x, c_y, c_z]^T$. The quantity q is a common denominator in all the expressions:

$$q = a_x(b_y - c_y) + a_y(c_x - b_x) + b_x c_y - b_y c_x$$

$$-qk = a_y b_z - a_z b_y - a_y c_z + a_z c_y + b_y c_z - b_z c_y$$

$$\begin{aligned} ql = {} & a_y \left(b_y^2 - a_y b_y - c_y^2 + a_x(c_x - b_x) + a_z(c_z - b_z)\right) \\ & + b_y \left(b_x(a_x - c_x) + b_z(a_z - c_z)\right) \\ & + c_y \left(a_y^2 + b_y c_y - b_y^2 + c_x(b_x - a_x) + c_z(b_z - a_z)\right) \end{aligned} \tag{3.22}$$

$$qm = a_x b_z - a_z b_x - a_x c_z + a_z c_x + b_x c_z - b_z c_x$$

$$\begin{aligned} -qn = {} & a_x \left(b_x^2 - a_x b_x - c_x^2 + a_y(c_y - b_y) + a_z(c_z - b_z)\right) \\ & + b_x \left(b_y(a_y - c_y) + b_z(a_z - c_z)\right) \\ & + c_x \left(a_x^2 + b_x c_x - b_x^2 + c_y(b_y - a_y) + c_z(b_z - a_z)\right) \end{aligned}$$

In terms of the equation system, the result is as follows, where $\boldsymbol{u} = [-k, -m, 1]^T$ and $\boldsymbol{v} = [l, n, 0]^T$:

$$\begin{gathered} \boldsymbol{u}^T\boldsymbol{u}d_3^2 + (2\boldsymbol{u}^T\boldsymbol{v} - \boldsymbol{u}^T(A + B))d_3 + (A^T B - \boldsymbol{v}^T(A + B) + \boldsymbol{v}^T\boldsymbol{v}) = 0 \\ d_1 = -kd_3 + l \\ d_2 = -md_3 + n \end{gathered} \tag{3.23}$$

Using the intermediate variables k, l, m, and n vastly reduces the size of the solution expressions compared to an explicit solution in terms of the coordinates of A, B, and C. Although it is possible to express the explicit solution for D, the ASCII text C-code version of it takes up almost 0.5 MB, i.e. close to 500,000 characters. Since the evaluation of a very large mathematical expression may be a problem for real-time applications, a small test has been made to see if the use of intermediate variables improves the execution time.

An implementation of both methods (with and without intermediate variables) has been made. For fully optimized code the average ratio of execution times for $100,000$ repetitions of solving the same system was 420. The resolution of the time measurements was 1 ms. This means that the introduction of intermediate variables has speeded up the process more than 400 times. I.e. the use of intermediate variables in this case is strongly recommended. Measured in milliseconds, the method with intermediates spent $6.0e-5$ ms on average for each repetition, whereas the full explicit solution spent an average of $2.5e-2$ ms on each repetition. The test was performed on a laptop computer running Windows Vista. The results are presented in Table 3.1.

Table 3.1: The results of the small-scale test of the consequences of using intermediate variables in the calculation of the solution to the non-linear system of equations. As the table shows, the result is that the execution time using intermediates is roughly 420 times faster than using the full explicit solution.

	Total execution time [ms]	Avg. time per repetition [ms]	Ratio (without/ with)
Without intermediates	2,516	$2.5e-2$	419.5
With intermediates	6	$6.0e-5$	

A Geometric View of the Solutions

As stated earlier in this section, the system of equations has 2 solutions. For this reason, it is useful to get a geometric interpretation of these two solutions. With some algebraic manipulations, the original system of equations can be changed as outlined in Equation 3.24.

This form of the system reveals that each equation of the system is geometrically the equation of a sphere. Thus, a solution to the system represents the intersections of 3 spheres. The centres of these spheres are the centre points of the edges of $\triangle ABC$. Since the edges of $\triangle ABC$ are face diagonals, this means that the sphere centres are located at the centres of three adjacent and orthogonal faces of the box to be constructed. The diameters of these spheres are equal to the length of the face diagonal that they are placed on. This implies that each sphere circumscribes the face that it is placed on.

The fact that the problem translates into the intersection of 3 spheres means that the solution to the system can be approached through the process of trilateration [34]. One version of trilateration[9] requires that the spheres are first transformed into a common coordinate system with one of the spheres at the origin and the other two located in one of the coordinate planes (e.g. $z = 0$). After doing trilateration, the solutions must be transformed back into the original coordinate system. The solution proposed above implicitly takes care of this transformation back and forth, since it makes no assumptions about corners A, B, and C other than $\triangle ABC$'s acuteness. Thus, the suggested solution is just another version of the trilateration process.

[9]Trilateration is commonly used for determining global positions in surveying and satellite navigation systems.

$$\left\{\begin{array}{l}(D-A)^T(D-B)=0\\(D-B)^T(D-C)=0\\(D-A)^T(D-C)=0\end{array}\right\}\Leftrightarrow$$

$$\left\{\begin{array}{l}(D-A)^T(D-B)+\left(\frac{(A-B)}{2}\right)^T\left(\frac{(A-B)}{2}\right)=\left(\frac{(A-B)}{2}\right)^T\left(\frac{(A-B)}{2}\right)\\(D-B)^T(D-C)+\left(\frac{(B-C)}{2}\right)^T\left(\frac{(B-C)}{2}\right)=\left(\frac{(B-C)}{2}\right)^T\left(\frac{(C-B)}{2}\right)\\(D-A)^T(D-C)+\left(\frac{(A-C)}{2}\right)^T\left(\frac{(A-C)}{2}\right)=\left(\frac{(A-C)}{2}\right)^T\left(\frac{(A-C)}{2}\right)\end{array}\right\}\Leftrightarrow$$

$$\left\{\begin{array}{l}D^TD-D^T(A+B)+\frac{(A^TA+2A^TB+B^TB)}{4}=\left\|\frac{(A-B)}{2}\right\|^2\\D^TD-D^T(B+C)+\frac{(B^TB+2B^TC+C^TC)}{4}=\left\|\frac{(B-C)}{2}\right\|^2\\D^TD-D^T(A+C)+\frac{(A^TA+2A^TC+C^TC)}{4}=\left\|\frac{(A-C)}{2}\right\|^2\end{array}\right\}\Leftrightarrow$$

$$\left\{\begin{array}{l}D^TD-2D^T\frac{(A+B)}{2}+\left(\frac{(A+B)}{2}\right)^T\left(\frac{(A+B)}{2}\right)=\left\|\frac{(A-B)}{2}\right\|^2\\D^TD-2D^T\frac{(B+C)}{2}+\left(\frac{(B+C)}{2}\right)^T\left(\frac{(B+C)}{2}\right)=\left\|\frac{(B-C)}{2}\right\|^2\\D^TD-2D^T\frac{(A+C)}{2}+\left(\frac{(A+C)}{2}\right)^T\left(\frac{(A+C)}{2}\right)=\left\|\frac{(A-C)}{2}\right\|^2\end{array}\right\}\Leftrightarrow \quad (3.24)$$

$$\left\{\begin{array}{l}\left(D-\frac{(A+B)}{2}\right)^T\left(D-\frac{(A+B)}{2}\right)=\left\|\frac{(A-B)}{2}\right\|^2)\\\left(D-\frac{(B+C)}{2}\right)^T\left(D-\frac{(B+C)}{2}\right)=\left\|\frac{(B-C)}{2}\right\|^2)\\\left(D-\frac{(A+C)}{2}\right)^T\left(D-\frac{(A+C)}{2}\right)=\left\|\frac{(A-C)}{2}\right\|^2\end{array}\right\}\Leftrightarrow$$

$$\left\{\begin{array}{l}\left\|D-\frac{(A+B)}{2}\right\|^2=\left\|\frac{(A-B)}{2}\right\|^2\\\left\|D-\frac{(B+C)}{2}\right\|^2=\left\|\frac{(B-C)}{2}\right\|^2\\\left\|D-\frac{(A+C)}{2}\right\|^2=\left\|\frac{(A-C)}{2}\right\|^2\end{array}\right\}$$

The spherical intersection interpretation of the equations explains the geometric relationship between the 2 possible solutions, D_1 and D_2, to the system. The following assumes that the two solutions exist.

The intersection of the two first spheres is a circle. The plane of this circle must be orthogonal to the plane of $\triangle ABC$, since the two sphere centres are both placed in the plane of $\triangle ABC$. Intersecting this circle with the third sphere produces two points on the circle. These points are illustrated in Figure 3.14.

The plane of $\triangle ABC$ bisects both the intersection circle and the third sphere since both their centres are placed in that plane. This fact combined with the orthogonality of the circle plane and the triangle plane means any intersections between the circle and the sphere must be symmetric around the plane of $\triangle ABC$. In other words, the two solution points, D_1 and D_2, are mirror images of each other through the plane of $\triangle ABC$. This also implies that the two boxes that can be constructed from points A, B, and C must be mirror images of each other. This is true because the trirectangular tetrahedra $ABCD_1$ and $ABCD_2$ each uniquely

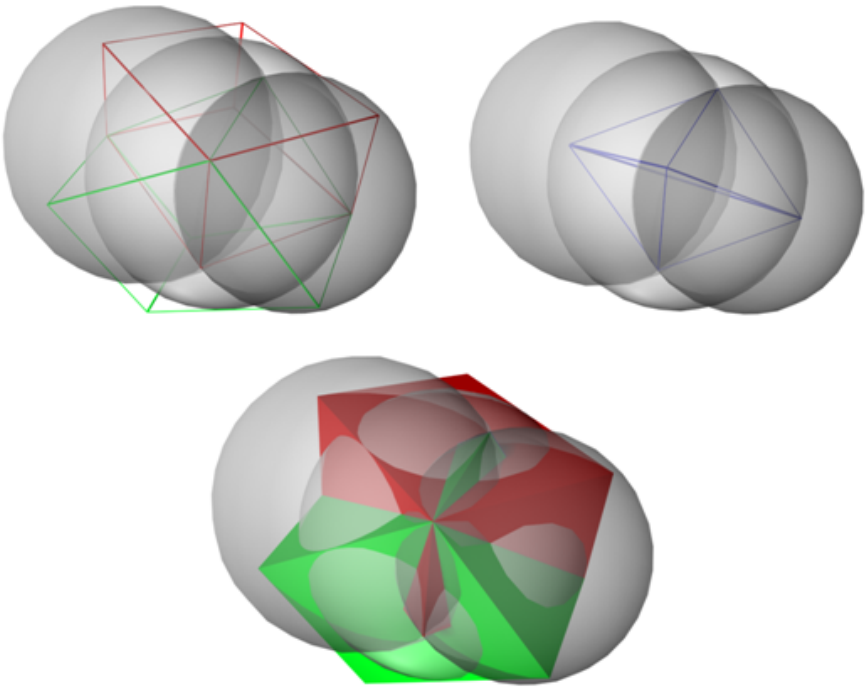

Figure 3.14: An attempt at illustrating the 3-D geometry of the system of equations. *(Top left)* The three intersecting spheres with the resulting boxes (red and green) overlaid. Note that the boxes are mirror images of each other. *(Top right)* The three intersecting spheres with the two resulting trirectangular tetrahedra (blue) overlaid. The intersection points correspond to the apices of the two tetrahedra. Triangle ABC is the base of both tetrahedra and can be seen in the centre of the figure. *(Bottom)* A rendering of the three spheres along with shaded versions of the boxes. The 2 solution points to the system are located at the box corners where the green box protrudes through the red box and vice versa. The plane of triangle ABC shows as a line, the axis of symmetry, between the red and green boxes.

defines a box with A, B, and C in common, but the D's mirrored through the plane of $\triangle ABC$.

One question remains about the system of equations: What is the geometric interpretation when there is less than two real solutions? It was already established that the system would have no solutions if the $\triangle ABC$ was obtuse, so this describes the case of no solutions. Having 1 solution would imply that that solution must be in the plane of the $\triangle ABC$ because of the mirror symmetry of the solutions. The only way that such a point can fulfil the orthogonality condition expressed in the equation system is if it is identical to A, B, or C, and that $\triangle ABC$ was a right triangle to begin with. Thus, the system always has two solutions, if the $\triangle ABC$ is verified to be acute before solving the system.

Disambiguating the Representation

The previous sections demonstrated that a representation of a box using the corners of an acute triangle produces two possible solutions for the complete box, thus making it a minimally ambiguous box representation[10]. Since the boxes are mirror images of each other, the ambiguity can be removed by fixing the allowable positions of D relative to $\triangle ABC$. This can be stated in several ways:

1. The winding order of $\triangle ABC$'s vertices when viewed from a solution point, D, must be fixed to being clockwise or counter-clockwise.
2. The handedness of a box coordinate system using one of the D's as origin and the directions towards A, B, and C as x-, y-, and z-axes must be fixed as right-handed or left-handed.
3. Choose the D which is formed with the positive or negative square root of the discriminant in the solution to the quadratic polynomial.
4. The sign of the matrix determinant, Δ_{ABCD_i} constructed from the box coordinate system axes must either be positive or negative:

$$\Delta_{ABCD_i} = \begin{vmatrix} a_x & a_y & a_z & 1 \\ b_x & b_y & b_z & 1 \\ c_x & c_y & c_z & 1 \\ d_{ix} & d_{iy} & d_{iz} & 1 \end{vmatrix} = \begin{vmatrix} a_x - d_{ix} & a_y - d_{iy} & a_z - d_{iz} \\ b_x - d_{ix} & b_y - d_{iy} & b_z - d_{iz} \\ c_x - d_{ix} & c_y - d_{iy} & c_z - d_{iz} \end{vmatrix} \quad (3.25)$$

If more information than the 3 points is available, e.g. the position of a user relative to the points, then this information may also be used for disambiguation.

It is worth noting that this mirror ambiguity is not the same as the one arising from an unknown handedness convention of the world coordinate system. Both solution boxes of this representation can be boxes in a right- or left-handed world coordinate system, no matter which internal constraints we add to the box.

An interesting prospect of constraining the mirror ambiguity is that it is possible to do so in a uniform way globally. This means that the constraint only needs to be defined and stored once - not for every box. If this is done, this representation becomes completely disambiguated.

[10]Minimally ambiguous, because anything less than 2 solutions is no longer ambiguous.

Finding the Remaining Corners

The remaining outstanding issue is to find the locations of the remaining 4 corners, E, F, G, H, of the box. The remaining corners and a visual representation of how to find them are illustrated in Figure 3.15.

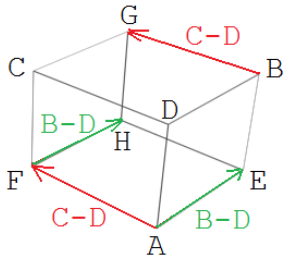

Figure 3.15: The four remaining corners of the box, E, F, G, and H need to be determined. The process of finding them is done by adding the green and/or red vectors as indicated by the figure. The same goal can be achieved in several other ways, but the general approach is the same: Addition of known edge vectors to a known corner.

Algebraically, the remaining corners can be determined as follows:

$$\begin{aligned} E &= A + (B - D) \quad F = A + (C - D) \\ G &= B + (C - D) \quad H = E + (C - D) \end{aligned} \tag{3.26}$$

Thus, the final outline of the algorithm for construction of a box from 3 corners, A, B, and C, forming a triangle of face diagonals is as follows:

1. Determine if $\triangle ABC$ is acute, otherwise terminate.
2. Find the 4^{th} corner, D, by solving the system of equations as outlined above.
3. Resolve the mirror ambiguity.
4. Use the direction vectors of the box coordinate system to determine the remaining corners.

Case II: Box Edge, Face Diagonal, and Box Diagonal

The case where $\triangle ABC$ is formed from a box edge, a face diagonal, and a box diagonal is the second case to be examined. Using the procedure from Case I as inspiration, the corners of the triangle are once again denoted A, B, and C. This is shown in Figure 3.16.

Figure 3.16 clearly demonstrates that a triangle formed from the edges prescribed by the case must be a right triangle. This is a simple consequence of the fact that the box is known to be rectangular. The fact that the triangle must be right means that the possible coordinates of C are highly dependent on the coordinates of A and B. This implies a loss of 1 DoF, because knowing just one of

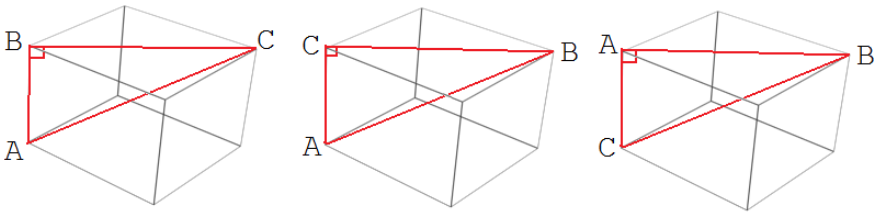

Figure 3.16: The box with the triangle of interest to this case indicated. The triangle corners have been named A, B, and C, respectively, for easy reference . The corners A, B, and C, have been defined in that order. The three cases cover the three options of how the order of the points relate to the different line segments on the box. *(Left)* A box edge has been defined first. *(Middle)* A box diagonal has been defined first. *(Right)* A face diagonal has been defined first. It is clear from the figure that the angle at B must be right if the box is rectangular.

the two acute angles of any right triangle, gives full knowledge of all three angles of the triangle. I.e. the 9 numbers of this representation may be expressed as 8 numbers without loss of information.

This means that the Case II representation will also be ambiguous, and to a larger extent than the Case I representation, because it suffers from loss of a DoF.

The questions to answer are the same as for Case I:

1. Where in 3-D space can a corner, C, be located relative to A and B?
2. How can a full box be constructed given A, B, and C?
3. How does the ambiguity manifest itself?

Constraints on Triangle ABC

The region where $\triangle ABC$ forms right triangles was already investigated during Case I. The results of this investigation are demonstrated in Figure 3.13. In brief, C must be picked from the surface of a slab perpendicular to AB, or from a sphere using AB as diameter. Which of the two surfaces to use, depends on where AB is located relative to the box.

Case I did not have any ambiguities at this level, because case I's $\triangle ABC$ was always acute, meaning that none of the three angles were special compared to the other two. In other words, all the edges of $\triangle ABC$ represented face diagonals, so none of them were special. Viewed in yet another perspective, the region where corner C formed acute triangles with A and B was a single, contiguous region, whereas the region of right triangles is split into two disjoint regions: A slab surface and a sphere surface, which forces an additional step to remove ambiguity.

If C is picked from the slab surface, then the relations between the box and the corners is as indicated by the left and right cases of Figure 3.16. The reason is

that in both cases, the right angle of $\triangle ABC$ is adjacent to the line segment AB, i.e. AB forms a cathetus of the right triangle.

If C is picked from the sphere surface, the picking sequence corresponds to the middle case of Figure 3.16. In the middle case, the right angle must be opposite AB, which means that AB forms the hypotenuse of $\triangle ABC$, i.e. a box diagonal.

Once the three corners are picked, we can freely assign new labels to them to reflect how they relate to the box instead of the sequence that they were picked in. This reassignment, however, requires that the picking order ambiguity has first been resolved. If we define corners P, Q, and R such that the angle at R is right and RQ is a box edge, then PQ is a box diagonal, and RP is a face diagonal. This is a more convenient relationship to use when determining a 4^{th} corner, D. The new assignment of labels is demonstrated in Figure 3.17.

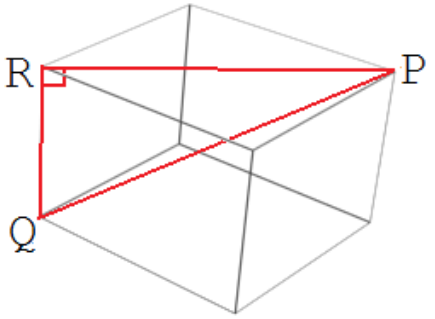

Figure 3.17: The three known corners with new names assigned. With the new labelling, the following is always true: At R, a box edge intersects a face diagonal at a right angle. At Q, a box edge intersects a box diagonal at an acute angle. At P, a face diagonal intersects a box diagonal at an acute angle. $\triangle PQR$ as a whole is therefore always a right triangle.

Determining the 4^{th} Corner

Finding the next corner, D, requires that it is not coplanar with $\triangle PQR$. This rules out one corner of the box, namely the one located at the other end of the box diagonal starting at R. The position of this coplanar corner, S, can easily be determined by adding the vector $Q - R$ to P:

$$S = P + (Q - R) \tag{3.27}$$

Adding S to the set of known points means that full knowledge about 2 box diagonals, 2 face diagonals, and 2 box edges, all in the same plane, is now given. This has two implications:

1. Our choice of D does not matter, since it will always be adjacent to exactly two known corners.

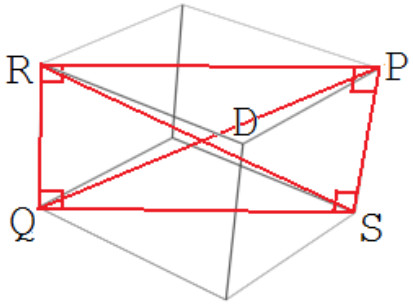

Figure 3.18: The corner S can straightforwardly be calculated from P, Q, and R. Using any combination of 3 of the 4 points P, Q, R, and S results in a right triangle with the same relation to the completed box. Its edges are always a box edge, a face diagonal, and a box diagonal. Since this is the case, the geometry of the equations governing the location of the 5^{th} corner, D, is both independent of the choice of D and the choice of triangle made from P, Q, R, and S.

2. All combinations made from 3 of the 4 known points form congruent, right triangles with the same relationship to the box. This means that any equation system set up using a combination of 3 of these corners is equally convenient, and result in the same geometry expressed in the system of equations.

These implications are illustrated in Figure 3.18.

As in Case I, the basis of the equations is the orthogonality of three pairs of vectors on the box surface:

$$\left\{ \begin{array}{l} (D-R)^T(Q-R)=0 \\ (D-R)^T(D-P)=0 \\ (D-Q)^T(D-P)=0 \end{array} \right\} \tag{3.28}$$

Before attempting to solve this system algebraically, it is useful to consider the geometry of the equations and their solutions. Any point, D, satisfying the first equation must necessarily lie on a plane through R with normal vector $\boldsymbol{n}_{\text{plane}} = (R-Q)$. The other two equations can be rewritten as sphere equations using the same method employed in Case I (see Equation 3.24):

$$\left\{ \begin{array}{l} (D-R)^T(R-Q)=0 \\ (D-R)^T(D-P)=0 \\ (D-Q)^T(D-P)=0 \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{c} (D-R)^T(R-Q)=0 \\ \left\| D-\frac{(R+P)}{2} \right\|^2 = \left\| \frac{(R-P)}{2} \right\|^2 \\ \left\| D-\frac{(P+Q)}{2} \right\|^2 = \left\| \frac{(P-Q)}{2} \right\|^2 \end{array} \right\} \tag{3.29}$$

The centres of these two spheres are the centre points of the implicated face diagonal, $(R+P)/2$, and the implicated box diagonal, $(P+Q)/2$. Furthermore, the radii of the spheres tell that they circumscribe the face defined by $\triangle PRD$ and

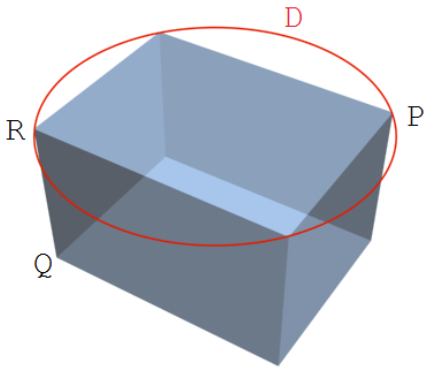

Figure 3.19: With three known points on a box, P, Q, and R, placed as indicated, the possible positions of D are indicated by the red circle, except R and P themselves. In this illustration, D is located where the red circle touches the box corners, revealing another ambiguity in the solutions for D. Thus, the illustrated box is but one out of an infinite number of possible boxes constructed from $\triangle PQR$.

the box as a whole, respectively. The intersection between these two spheres is a circle. The normal of the plane of this circle, $\boldsymbol{n}_{\text{circle}}$, can be found by subtracting the sphere centres:

$$\boldsymbol{n}_{\text{circle}} = \frac{R+P}{2} - \frac{P+Q}{2} = R - Q = \boldsymbol{n}_{\text{plane}} \tag{3.30}$$

Thus, the normals of the circle plane and the plane of $\triangle PRD$ are the same. Furthermore, substituting the corners R and P into the sphere equations reveals that they both satisfy both equations, thus being part of the intersection circle. This means that the intersection circle circumscribes $\triangle PRD$. This in turn implies that the solutions to this equation system form a circle in 3-D space. This circle has centre point $(R+P)/2$ and radius $\|R-P\|/2$, yielding the following equations for D:

$$(D-R)^T(R-Q) = 0 \quad \wedge \quad \left\| D - \frac{(R+P)}{2} \right\|^2 = \left\| \frac{(R-P)}{2} \right\|^2, \quad D \notin \{R, P\} \tag{3.31}$$

The circular shape of the solution region is also completely coherent with the fact that any solution point, D, must form a right triangle with corners P and Q. The only exceptions are, as usual, that D is neither allowed to be identical to R nor P. A box with its solution circle for D is illustrated in Figure 3.19.

Disambiguating the Representation

The unfortunate part of determining that the solution points form a circle is that the ambiguity in this representation offers an infinite number of possible boxes.

These boxes will all have different *interdependent dimensions and orientations*. Removing the ambiguity must in all cases involve deciding if D lies to the left or right when looking from R towards P using $\boldsymbol{n}_{\text{circle}}$ as up-vector. This part of the ambiguity is equivalent to the mirror ambiguity found in Case I.

Having dealt with the mirror ambiguity, the remaining ambiguity can be handled by:

1. Fixing the length of one of the line segments RD or PD. This length must be in the open interval $]\,0\,;\,|RP|\,[$.
2. Fixing the angle between RD and RP (or alternatively between PD and PR).

Once D is found, the remaining 3 corners can be found using a method analogous to the one employed in Case I. This concludes the investigation of Case II.

Case III: Two Box Edges and a Face Diagonal

In this case, all three given corners, A, B, and C, lie on the same face of the box. This leads to two immediate conclusions:

1. The triangle, $\triangle ABC$, must be right, since any two adjacent box edges meet at a right angle. Thus, like Case II, the representation has 1 DoF less than the minimum of 9.
2. All lying on the same face of the box, the corners of $\triangle ABC$ provide no information about one dimension of the box. This means that no information about its size in this dimension is present in the available data.

A box along with the given points of $\triangle ABC$ is illustrated in Figure 3.20, where it is clear that no information about the third dimension of the box is given by $\triangle ABC$.

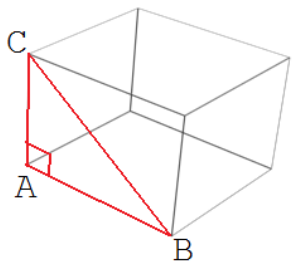

Figure 3.20: A box with the known corners, A, B, and C, indicated. The figure shows that all three corners of $\triangle ABC$ lie on the same face of the box. This means that $\triangle ABC$ must be a right triangle and that its corners give no information about the box dimension orthogonal to the plane of the triangle.

The case of picking points A, B, and C for a right triangle was already investigated in section 3.4. In brief, the conclusion of this investigation revealed that the sequence of mapping the three corners mattered. This means that some ambiguity must be dealt with already at the time of definition of $\triangle ABC$. This ambiguity is dealt with by deciding if AB is going to define a box edge or a face diagonal.

Determining the 4^{th} Corner

Even though the given triangles are right both in Case II and Case III, the process of deriving the next corner from the available data is different because of the different mapping of triangle corners to box corners. The next corner, D, must, as in the previous two cases, be non-coplanar with the $\triangle ABC$ to be of any interest. In this case, this implies that D must be on the face opposite $\triangle ABC$. If we choose D such that $ABCD$ forms a trirectangular tetrahedron with A as the apex (illustrated in Figure 3.21). This yields the following linear equation system:

$$\left\{ \begin{array}{l} (D-A)^T(C-A)=0 \\ (D-A)^T(B-A)=0 \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{l} D^T(C-A)-A^T(C-A)=0 \\ D^T(B-A)-A^T(B-A)=0 \end{array} \right\} \tag{3.32}$$

The solutions to this system form a line in space. Instead of solving the linear system directly, the equation of the solution line can be derived in a, perhaps, more illuminating manner through geometric arguments:

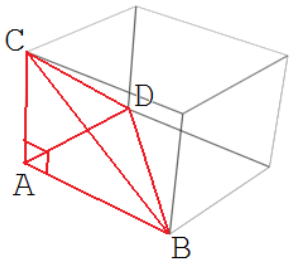

Figure 3.21: The unknown point, D, is added to the box such that $ABCD$ forms a trirectangular tetrahedron.

The corner D must, necessarily, be a point in both the planes of triangles $\triangle CAD$ and $\triangle BAD$. Thus, D must be found in the intersection between these two planes. Furthermore, the box geometry dictates that these planes are orthogonal to each other, which implies that the planes always intersect each other somewhere. The intersection between any two non-parallel planes is a line in space. This means that D must be found somewhere on this line. The same goes for A, since A is present in both $\triangle CAD$ and $\triangle BAD$. The known corner A can therefore be used as a fixed point on the line. The direction of the line must be orthogonal to both line segments AB and AC. This direction can be computed as the cross product of vectors $\boldsymbol{u} = (B-A)$ and $\boldsymbol{v} = (C-A)$. The parametric equation of the solution line, $D(\lambda)$, then becomes:

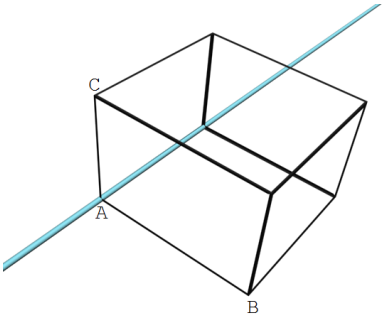

Figure 3.22: A box with the solution line for D, given $\triangle ABC$. The shown box is therefore only one of an infinite number of possible boxes constructed from triangle ABC. All possible boxes are non-uniformly scaled versions of the one shown (scaled along the direction of the solution line).

$$D(\lambda) = A + \lambda(\boldsymbol{u} \times \boldsymbol{v}), \quad \lambda \neq 0 \tag{3.33}$$

This line is illustrated in Figure 3.22.

Disambiguating the Representation

The fact that D can be anywhere (except at A) along a line means that an infinite number of boxes of different non-uniform scale can be constructed from the representation. The first level of ambiguity is the direction vector of the line $D(\lambda)$. Depending on the relative positions of points A, B, and C, the spatial direction of the cross product vector can be inverted. This ambiguity is analogous to the mirror ambiguity in the two previous cases, and must be settled before the scale ambiguity can be resolved. The suggested means for solving the scale ambiguity are:

1. Settle the desired length of the box in the missing dimension (along AD) directly.
2. Fix the value of λ in the equation $D(\lambda)$.

If the decision is to fix the value of λ to λ_0, then it is useful to know the length of the box edge going from A to $D(\lambda_0)$. This length can be derived using a couple of properties of the cross product operator:

$$\|D(\lambda_0) - A\| = \|(A + \lambda_0 \boldsymbol{u} \times \boldsymbol{v}) - A\| = |\lambda_0| \|\boldsymbol{u} \times \boldsymbol{v}\| = |\lambda_0| \|\boldsymbol{u}\| \|\boldsymbol{v}\| \sin\theta \tag{3.34}$$

In the present case, the angle between the vectors, θ, is known to be $90°$, which reduces the expression above to:

$$\|D(\lambda_0) - A\| = |\lambda_0| \|\boldsymbol{u}\| \|\boldsymbol{v}\| \tag{3.35}$$

The remaining corners of the box can now easily be found using the methods outlined for the same purpose in Case I.

Summary of 3-Corner Representations

Having investigated Cases I-III, all possible box representations involving 3 box corners have been treated. The main results of this investigation were that all of the representations were ambiguous. However, Case I was only minimally ambiguous, representing only 2 possible boxes. Furthermore, the relation between the 3 corners of Case I is highly symmetric. These observations make Case I more attractive to use as a box representation than Case II and Case III. In each case, some possible methods for resolving the ambiguity was presented. The results are summarised in Table 3.2.

Table 3.2: Summary of the 3 possible representations of boxes using 3 box corners. The Case I representation holds the most promise as a direct basis of a box shaping technique, since it is less ambiguous than the others. The ambiguity region is the region of 3-D space from which the 4^{th} box corner, D can be picked, once $\triangle ABC$ is known.

	$\triangle ABC$ edge-box relation	$\triangle ABC$ type	DoF	Number of boxes	Ambiguity region for D
Case I	3 face diagonals	Acute	9	2	2 points
Case II	1 box edge, 1 face diagonal, 1 box diagonal	Right	8	∞	Circle
Case III	2 box edges, 1 face diagonal	Right	8	∞	Line

The mirror ambiguity present in all three cases can be fixed rather elegantly by imposing a global constraint on the internal handedness of box coordinate systems.

Although ambiguous in different ways, all of the three cases can be used as the basis of 3-D box shaping techniques, since they all require the user to input very little data. If the representations are used as practical box shaping techniques, then some disambiguation steps must be incorporated into the techniques.

3.5 Other Minimal Representations of a Box

A minimal representation of the box is one that features exactly enough parameters, i.e. 9, to unambiguously define a box. These 9 parameters must settle the position, orientation, and dimensions of the box. As stated earlier, any such minimal representation can only be unambiguous if there is an a priori settlement of coordinate system handedness.

Unlike the exhaustive survey of 3-corner representations, this section will not seek to exhaust all possible minimal representations, since there does not seem to be any systematic way of covering all the possibilities. In other words, there is a plethora of ways of encoding the 9 independent parameters as geometric shapes and quantities that can be converted into a box.

As examples from this plethora of possible interpretations of data, information given as a point and a direction vector can be interpreted as: 1) A position and an angle-axis rotation, 2) a plane and a box dimensions, 3) a circle in 3-D space, etc. The list of geometric interpretations of the parameters is endless.

Rather than attempting to cover all options, a couple of simple and straightforward representations will be examined. Other minimal representations must somehow encode the same information as the two suggested below.

Position, Orientation, and Scale

Possibly the simplest way of defining a box, is to directly give the parameters of its 9 DoF: A 3-D position, a rotation vector or 3 Euler angles, and three scaling factors. The 3-D position most conveniently maps to the position of the box centre. This representation can readily be used to transform an axis-aligned, origin-centered cube[11] into a box of any position, orientation, and dimension. An illustration of this approach to box construction is shown in Figure 3.23. The procedure can be expressed mathematically as follows:

$$\boldsymbol{p}_{\text{box,world}} = \boldsymbol{T}_{x,y,z}\boldsymbol{R}_{\boldsymbol{r},\theta}\boldsymbol{S}_{s_x,s_y,s_z}\boldsymbol{p}_{\text{std.cube,local}}, \quad s_x, s_y, s_z \neq 0 \tag{3.36}$$

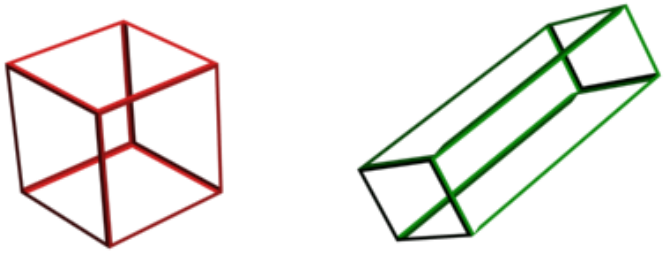

Figure 3.23: The standardized, red cube is transformed into the desired green box using the parameters given in the box representation.

Using the centre point of the box as the position given is referred to as the most common representation for oriented bounding boxes (OBBs) in [14].

[11] This standard cube always has the following corners: $[-1,-1,-1]^T$, $[-1,-1,1]^T$, $[-1,1,-1]^T$, $[-1,1,1]^T$,$[1,-1,-1]^T$, $[1,-1,1]^T$,$[1,1,-1]^T$,$[1,1,1]^T$. The standard cube is identical to the cube representing the viewing frustum in the OpenGL normalized device coordinate (NDC) space.

Box Diagonal and Orientation

In this representation, a box diagonal and the box orientation is given. Knowing the box orientation means that the directions of the axes of the box coordinate system are known. Full knowledge of the box diagonal can be practically represented in at least two ways, each totalling 6 independent parameters:

1. Both end points, A and B, of the box diagonal.
2. One box corner, A, and a direction vector, $\boldsymbol{d} = B - A$,

Full information about the box diagonal together with orientation implies that both box dimensions and position can be extracted. In other words, the remaining corners of the box can be determined by the following procedure:

1. Get the box rotation matrix by transforming the box orientation information into matrix form.
2. Determine the box dimensions for use in the scale transformation matrix by projecting $\boldsymbol{d}$ onto the columns of the orientation matrix.
3. Use the mid-point, C, of the line segment AB, i.e. $C = (A + B)/2$, as translation vector in the box translation matrix.
4. Apply the transformation matrices determined in the above steps to the corners of the standard cube.

3.6 Non-minimal Box Representations

This final section on box representations treats some of the many possible ways of representing a box using more than the required 9 independent parameters. Using more than 9 parameters means that the information offered by the extra parameters will be redundant. The representations suggested are by no means an exhaustive list, but introduces the ones suggested in [14] as OBB representations, as well as a couple of others.

Three Face Diagonals and Local Handedness

This representation is the disambiguated version of the one described in section 3.4, where three face diagonals are given. The representation contains the following information:

1. Three points, which form an acute triangle. This triangle's edges form connected face diagonals on the represented box.
2. The desired local handedness of the box coordinate system. Can be expressed as a matrix determinant as shown in section 3.4.

This representation needs 10 parameters to be explicitly defined. If the local handedness is the same for all boxes, then the number of parameters needed for each box is reduced to 9, making it a minimal representation as was discussed in section 3.4.

One Corner, Two Box Edges, One Scale Factor, and Local Handedness

This representation is essentially a disambiguated version of the one investigated in section 3.4, also called Case III for 3-corner representations of boxes. The representation consists of:

1. A known box corner.
2. The direction vectors of the two edges meeting at the given corner. Alternatively, two adjacent corners to the first one given.
3. The size of the missing box dimension. This disambiguates the scale of the box.
4. The mapping from direction vectors to local box coordinate system axes. This disambiguates the mirror ambiguity.

This representation disambiguates Case III by adding a scale factor, the size of the box in the missing dimension. The data for this representation requires the explicit definition of 11 parameters.

Four Non-Coplanar Corners

In this representation, 4 non-coplanar corners are given. The 4-corner representations of a box can also be regarded as disambiguated extensions of the 3-corner representations presented in section 3.4.

There are four distinct ways of relating the corners of a tetrahedron to the corners of a box. All these four possibilities are shown below in Figure 3.24.

The procedure for finding the remaining corners in any of the 4 cases was outlined in section 3.4. Depending on the tetrahedron-box relationship encountered, the specific procedure changes slightly. However, the problem of finding the remaining corners is always solvable by simple vector arithmetic, only using the given points and vectors as input.

This representation requires 12 parameters no matter which tetrahedron type is given.

Origin, Three Axes, and Three Scale Factors

This representation can visually be thought of as defining the origin of a box-aligned coordinate system along with its axes expressed in a world coordinate system, and the individual scale factors for these axes. The scale factors of the axes can then be directly translated into the dimensions of the box. The data of this representation is:

1. A 3-D point defining the origin of a box-aligned coordinate system.
2. 3 orthonormal vectors or, equivalently, a rotation matrix giving the axes of the box.
3. 3 scale factors (box dimensions).

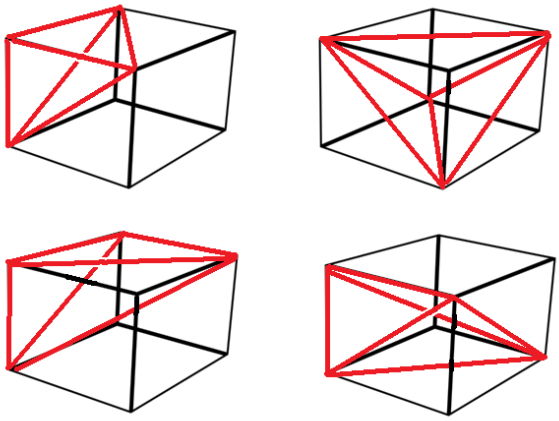

Figure 3.24: All 4 distinct ways of picking 4 non-coplanar corners from a box. All but the bottom left case can be produced by adding an extra corner in the 3 face diagonals representation presented in section 3.4. Note that none of the red tetrahedra have any obtuse angles. *(Top left)* A trirectangular tetrahedron formed from 3 face diagonals and 3 box edges. *(Top right)* A tetrahedron formed from 6 face diagonals. In a sense, this is the natural extension of the triangle representation formed from 3 face diagonals, since both are highly symmetric wrt. the triangle edge-box relationships. *(Bottom left)* A tetrahedron formed from 2 face diagonals, 3 box edges, and 1 box diagonal. *(Bottom right)* A tetrahedron formed from 3 face diagonals, 2 box edges, and 1 box diagonal.

The origin of this representation is most conveniently placed either at a box corner, or in the geometric centre of the box . If a corner is chosen as origin, this representation is equivalent to defining a trirectangular tetrahedron. If the geometric centre is chosen as origin, then the representation is equivalent to the one presented in [14] as the most common and convenient representation of an OBB for collision detection purposes.

Since the direction of a single axis only has 2 DoF, one can either choose to represent the box dimensions implicitly as the length of each axis, or to express scale as an explicit quantity. Depending on this choice, the representation requires 12 (implicit scales) or 15 (explicit scales) parameters.

The source of redundancy in the representation is the 9 parameters of the rotation matrix instead of the minimum 3.

Six Planes

This representation uses the notion that a general box can be thought of as the volume contained inside the intersections of 3 orthogonal pairs of parallel planes.

To define 6 planes, each plane must have a distance from the origin (1 DoF) and a normal vector (2 DoF), producing a total of 18 parameters.

Three Slabs

The number of parameters for 6 planes can be vastly reduced by defining 3 mutually orthogonal slabs instead. 3 slabs reduce the redundancy by using the same normal for parallel planes, and encoding the positions of half the planes as slab widths.

One slab requires an origin distance, a normal vector, and a width. Depending on whether normal and width are represented explicitly or are combined into 3 numbers, the number of parameters for a single slab becomes 4 or 5, respectively.

The grand total of parameters for 3 slabs then becomes 12 or 15. The redundancy of the representation mainly comes from the fact that the necessary orthogonality of the slabs is not made implicit in the representation.

Eight Corners

The final representation to be considered is to define all 8 corners of a box explicitly. This representation is, as stated at the very beginning of this chapter, redundant, since there are several constraints on the possible relative locations of 8 points, if they are to form a rectangular box. The representation is, however, very straightforward to understand, and to render graphically.

To make this representation useful, some convention must be settled about how the 8 corners are to be connected to form the faces and edges of a box. Alternatively, using the coplanar property of corners on the same face, the connectivity of the corners can be determined automatically.

The number of parameters in an 8-corner representation is 24.

3.7 Box Geometry and Representation Summary

In this chapter, the basic geometry of 3-D boxes has been introduced along with a number of possible computational representations of this geometry.

The main results to note in this chapter are twofold: First of all, that an arbitrary 3-D box has 9 DoF, i.e. position, orientation, and non-uniform scale. These DoF translates into 9 parameters, which must given in any computational model of the box. This fact also settles the basic rules that *any* 3-D box shaping technique must adhere to. The second result to note is that the 9 DoF can be represented in a multitude of ways, some of which do not explicitly state position, orientation, and scale. This observation forms the basis of interesting possibilities of designing 3-D box construction techniques for users of VE's.

One particular representation, the one based on defining 3 face diagonals of a box through the corners of an acute triangle, is used in all of the papers contributed by this thesis. In the first paper, box shaping techniques based on acute triangles are introduced and compared to existing techniques. In the second paper, one of the techniques is used in getting quantitative results about users' ability to recognize box shaped patterns in an immersive virtual environment. In the final paper, the same technique is used and evaluated as the basis of a 3-D box shaped selection tool.

4 Summary of Contributions

4.1 Introduction

This thesis presents its main contributions in the form of three papers. Each paper is based on a proposed solution to an interaction problem, which somehow involves the definition of volumes of space. Furthermore, the papers use the results of the previous papers in a sequential manner, such that the lessons learned in one paper has a direct bearing on the approach of the next paper. The papers are all structured in a specific pattern. First, the addressed problems are introduced and motivated. This includes a discussion of relevant, current state-of-the-art within each paper's particular area of focus. Based on these discussions, new technique(s) are proposed to solve the problem, and subsequently a formal evalutation of the proposed technique(s) in the form of a user experiment is described and analyzed.

The focus of the first paper is the definition of box-shaped volumes of space, without any particular associated application. This means that box-shaping is viewed as an interesting discipline in itself, with many possible applications, as described in chapter 3. The basis of this paper is the novel idea of defining a box in terms of 3 of its corners. A series of box construction techniques based on this idea are designed and evaluated in the first paper. Since the evaluation of the techniques indicates that 3-corner (3C) techniques are superiour to existing techniques, 3C techniques play important roles in both of the subsequent papers. Another result from the first paper, which finds use in the second paper is the introduction of a box error metric, which allows quantitative measurement of the difference between two boxes.

The second paper focuses on two problems related to the perception of volumes of space, in the case where a volume is visualized by 3-D objects, or *glyphs*, which represent a discrete sampling of that volume of space. The two problems addressed are: 1) The visual occlusion problems caused by other objects in the direct vicinity of the volume of interest, and 2) The problem of the density of the sampling, which is required for perceiving the shape of such a volume correctly. The paper introduces a technique for rendering glyphs, called constant visual angle (CVA) glyphs, which has the potential of dealing with occlusion problems. The experiment uses box shaped volumes, and the 3C technique as user task, in order to quantitatively measure how accurately the presented volumes are perceived. Subsequently, CVA glyphs and the knowledge gained about

density requirements are used in the studies of the third paper.

The third paper investigates a specific application of volume definition, namely the task of multiple object selection (MOS). The specific area of interest is multiple object selection on a large scale, i.e. when there are hundreds or thousands of objects to be selected. The main parameters of interest are the choice of MOS tool, and the geometric layout of the objects to be selected. Three different MOS techniques have been tested, among which is the magic wand technique, a novel technique in a 3-D MOS context. Another tested technique is one of the 3C box shaping techniques. Thus, the third paper builds on the results of the former two papers.

4.2 Paper A Summary

Introduction

Paper A is entitled *Shaping 3-D Boxes: A Full 9 Degree-of-Freedom Docking Experiment*. The overall goal of Paper A is to identify efficient techniques that allow a user to shape a 3-D box with all 9 associated DoF. The reason for aiming a study at the specific interaction of shaping boxes is the ubiquity of boxes both in 2-D user interaction, and in real-world objects. I.e. the need to shape a box has the potential to be an important part of many interactions and applications, such as interactive modelling and in volumetric selection tasks. The goals of the paper are twofold. First of all, a new family of techniques based on pointing out 3 corners of a box is introduced and tested. The second goal is to see, how various allocations of the 9 box DoF to the users' hands affects the quality of the resulting boxes.

The basic outline of the paper is to first identify the current state-of-the-art in 3-D box construction methods. Then, the option to avoid direct specification of orientation and scale is proposed, and the theory of this approach, which is based on indicating 3 corners of a box, is introduced. This theory is basically the same, which was also given in somewhat greater detail in chapter 3. The evaluation of the new technique needs a way of quantifying the difference between a user created box and a reference box, called the target. For this reason, a quantitative measure of box difference is also introduced in the paper. The final portion of the paper deals with the performed experiment, and the analysis of its results.

State-of-the-Art

Shaping 3-D Boxes

Shaping 3-D boxes in immersive virtual environments had not been the subject of much study prior to the paper presented in this thesis. This is especially the case, if the box construction technique is detached from a particular application such as object selection. As such, the paper presented in this thesis is currently both unique in its theoretical and experimental approach to 3-D box construction. In this section, the main current examples of box construction techniques will be presented.

The main existing work in the field of 3-D box construction is due to Ulinski et al [60, 59, 61, 58]. The focus of these studies is the application of 3-D boxes as selection volumes. Outside the world of desktop 3-D box construction, these works present the first systematic studies of 3-D box construction techniques. The approach taken by Ulinski et al is to study how the actions of the user's hands can be mapped to the parameters controlling the shape of a box. Specifically, techniques involving both hands, i.e. bimanual techniques, are of interest.

In the studies, three families of techniques are introduced and tested as selection tools: Hand-on-Corner (HOC), Hand-in-Middle (HIM), and Two Corners (TC). The techniques are designed such that they feature different levels of the two primary design factors of synchronicity and symmetry between the actions of the hands. Synchronous in this respect implies that the hands are performing tasks at the same time, while symmetry means that the hands are performing the same task.

A HOC technique is one, where the user's non-dominant hand holds onto one corner of the box being shaped. This hand is both used for placing that corner of the box, and for orienting the box as a whole. The dominant hand places the diagonally opposite corner to give the scale of the box. Thus, HOC techniques are all asymmetric, since the hands do not play the same role wrt. the box. The synchronicity of a HOC technique can be modified by deciding whether or not the hands perform their tasks simultaneously or not. The fact that position, orientation, and scale are all defined separately as explicit quantities, means that a HOC technique features all needed 9 DoF to define a full box. A HIM technique is more or less identical to a HOC tehnique, except for the fact that the non-dominant hand defines the geometric centre of the box, rather than a corner of the box. Thus, a user employing HIM techniques can potentially create a box twice as large, since the arms only define half the box diagonal, instead of the full length of the same diagonal.

In techniques based on the TC idea, the user still holds on to two diagonally opposite corners of the desired box. However, with the basic TC scheme, the positions of the two hands are used to infer both the position, orientation, and scale of the box. The main appeal of this approach is that it is simple. As such it also performs well in evaluations. However, TC is problematic, since two 3-D positions, i.e. 6 numbers, cannot possibly produce 9 *independent* parameters. This implies that some internal dependency between the computed parameters occurs. In box shaping terms, this means that some boxes will be impossible to define. In terms of DoF, it implies that the orientation and scale of the box become interdependent, i.e. modifying the scale of a box also changes its orientation and vice-versa. Another possible result is that the created boxes are all axis-aligned, i.e. of identical orientation. An attempt to fix this problem is presented in [58]. The presented solution is to allow the user to rotate the box around the diagonal defined by the user's hands, by twisting the hands. This adds an additional DoF to the technique. A scale button is also included, which adds an 8th DoF. However, the technique is still missing 1 DoF, meaning that it is in some way ambiguous. The implications of this are twofold: 1) Box shaping algorithms based on 3 corners are still attractive, because two corners are insufficient, unless a series of extra steps or controls are added. 2) TC techniques are not considered in

the current evaluation of 9 DoF box shaping techniques. Thus, all comparisons will be made against HOC type techniques, which feature all needed 9 DoF. Furthermore, both the HOC techniques and those introduced in the presented contribution are based on controlling corners, rather than the centre point of a box.

Another approach to shaping a 3-D box is presented by Lucas et al in [33, 32]. The application is focused on resizing objects [33], i.e. a kind of non-rigid manipulation, and on selection of multiple objects [32]. The approach taken is to shape the box from a persistent proto-box, which can be arbitrarily moved, oriented, and scaled using separate techniques. In [32], the positional and rotational aspect of manipulation is handled using the Go-Go technique [46]. However, the scaling is handled by aligning a cursor with the desired dimension of the box, followed by pressing a button or joystick to perform the scaling. This technique is named the Pointer Orientation-Based Resize Technique, or PORT. In [33], an approach to scaling using 26 separate widgets spread out across the box is used. With that approach the scaling occurs, when the user selects and moves a widget. The chosen widget decides which dimensions are scaled. Thus, the presented techniques in those papers are not full-fledged 9 DoF box shaping techniques, but rather ensembles of techniques that allow for serial manipulation of the same DoF as those of a box.

In standard 3-D modelling tools for the desktop like 3ds Max, an arbitrary box is also modelled in a series of steps. The first step is to define an axis-aligned base of the desired box. This step settles two of the scale DoF. The next step is to define the 3rd scale DoF, which is done by extruding the base along the missing dimension. The final two steps is to orient and place the box, which settles the remaining 6 DoF in two operations.

Since the publication of Paper A, an additional study has emerged, which uses the non-rigid manipulation of a small box as interaction technique for touch displays [9]. The presented tBox technique is specifically aimed at touch displays, which means that is not directly usable in an immersive HMD context like the one used in this thesis. However, it is a very good example that the ability to control the DoF of a box has many applications besides the actual shaping of the box itself.

Docking

The performed experiment is a *docking* experiment. This means that the user task is to match the spatial characteristics of a user-shaped box to those of a presented target box in-situ. The rationale of using a docking task is that this task is very generic, and not tied to any specific application of box-shaping. However, being able to make precise, high-quality dockings is essentially the same as performing a high precision volumetric selection, non-rigid manipulation, or modelling task. As such, results from a docking experiment should be indicative of the performance in several application contexts.

No prior studies performing a docking task using a non-rigid 9 DoF object exist. However, many studies exist, if the restriction is 6 DoF or lower. Prominent examples are presented in [38, 74, 72, 35, 21, 23]. Docking tasks have typically been used to either evaluate the efficiency of interaction techniques related to

rotation manipulations [35, 21], or to evaluate the efficiency of novel interaction devices [23, 74, 72]. The main results of these works can be summarized in two points.

1. Users tend to focus on one or a few DoF at a time when doing a docking task.

2. Rotation is harder to control precisely than translation.

The latter point implies that an interaction technique which implicitly achieves arbitrary object rotation should outperform any technique involving explicit rotation. This is exactly one of the defining characteristics of the techniques introduced and tested in Paper A.

While none of the existing studies deal with docking tasks involving more than 6 DoF, one of the studies does introduce and test interaction devices with more than 6 DoF [23]. In that study, 12 DoF input devices are evaluated in docking tasks. However, the additional DoF are used such that several device DoF maps to the same DoF in the visualized environment, e.g. such that there is more than one way of controlling rotation.

Measuring Docking Quality

Measuring the quality, or specifically precision, of a docking operation is a non-trivial task. The only current way of measuring docking quality is due to Zhai [74]. The proposed quality measure, q, deals with the notion of inefficiency in the docking process, in that it is based on the observed path length, l, and the theoretical minimum path length, m, as given in Equation 4.1. The equation for the rotation component of the docking process follows a similar pattern.

$$q = \frac{l - m}{m} \tag{4.1}$$

Since Zhai's main interest was to investigate how users allocate control of the 6 DoF used in the experiments, the measure introduced deals with the docking process rather than its end results. Furthermore, the measure is split into two separate components for translation and rotation. In this paper, however, the goal of the evaluation is not to investigate the path that leads to the docking result, but rather how close the end result was to the given docking target. This is very much in line with the methodological decision of this thesis to allow for imperfect results in the user experiments. No other current work has attempted to quantify the end results of a docking operation, since docking tasks have typically been required to be perfectly solved by the user, i.e. the result is always within some very close, acceptable proximity of the target shape.

The only other measurement frequently used in current literature on docking is the completion time of the task, and the time spent controlling each of the DoF involved. Completion times, however, are not specific to docking tasks, and furthermore, they say little about the actual quality of the end result.

In general literature dealing with boxes, such as Ericson's book on collision detection [14], the only proposed methods for comparing two boxes deal with

the questions of overlap and separation/penetration distances. As such the presented methods do not deal with measuring the similarity of boxes.

For these reasons, the method of measuring the similarity of two boxes, and hence the docking quality, presented in this paper is both necessary and novel. The method is both explained in the next sections, and in the paper itself.

Contributions

The main contributions of this paper fall into a technical category and an experimental category. On the technical side of matters, the introduction of new box shaping techniques and a metric to compare the similarity of two boxes are the main contributions. On the experimental side, the validation of the proposed box shaping techniques and their comparison to existing techniques are the main contributions. In this section, these contributions will be introduced in turn.

3-Corner Box Shaping Techniques

The theoretical background of defining an arbitrary 3-D box by defining three of its corners was already presented in chapter 3. However, the theoretical possibility of using 3 corner representations, henceforth 3C representations, as the foundation of a box shaping technique was not explored then. In Paper A, four different techniques were designed from the basic 3C concept. All of these techniques are based on letting the user point out 3 corners of the desired box. As stated in chapter 3, these three corners must be placed such that they form an acute triangle whose edges are face diagonals of the desired box. Therefore, the basic requirement for all techniques is that the user has to be able to point out three 3-D locations in space. This requirement is handled by a 3-D motion tracking system and a couple of interaction devices with buttons, which allows the user to indicate where to place the corners.

The main distinguishing feature of box shaping techniques based on 3C thus lies in the number of corners placed simultaneously. This translates directly into the number of box DoF, which the user has to simultaneously control. I.e. placing one corner requires control of 3 DoF, two simultaneous corners require 6 DoF, and three simultaneous corners implies control of 9 simultaneous DoF. Although it is also possible to control less than 3 simultaneous DoF, e.g. by controlling the first, second, and third coordinates of a corner position in separate steps, the choice here is to only work with DoF control in batches of 3. The reasons for this choice are:

1. With a 6 DoF motion tracking system such as the one used in the current experiment, it intuitively seems more natural to measure all 3 coordinates at the same time when a user is pointing somewhere, than to tell the user to first indicate the x-coordinate, then y, then z.

2. Splitting the DoF definition into smaller batches than 3, produces a large number of possible combinations to test, which is unmanageable given the available time and resources.

Therefore, with the chosen DoF batches of 3, 6, and 9, the possible 3C box shaping techniques are:

1. **3+3+3 DoF 3C**, which means that the user places the three corners, one at a time. The main implications of this strategy are that the user only has to use one hand at a time, and that the number of simultaneously controlled DoF are minimized.

2. **3+6 DoF 3C**, which means that the user first places one corner, then two simultaneously, to define a box. The final step may also be expressed as simultaneously defining both end points of a face diagonal. Thus, this technique mixes the DoF batch sizes.

3. **6+3 DoF 3C** is the opposite procedure of 3+6 DoF 3C. I.e. the user first defines the two end points of a desired face diagonal simultaneously, then places the final corner to complete the box. This technique also mixes the DoF batch sizes.

4. **9 DoF 3C** is special, since placing three corners simultaneously implies that two hands are no longer enough. As such, this technique is somewhat complicated to control, and also somewhat contrived in its design. However, it provides a counterpart to the original HOC technique, where all 9 DoF are also controlled at the same time. In the current implementation, control of the third corner is allocated to a position offset in front of the user's head.

Since the 3C representation is novel, all of the above techniques are also novel. The main reason for them also being potentially useful is the lack of requirement of explicit orientation specification, which has proven difficult in numerous previous studies.

Computing the Similarity of Boxes

As argued in section 4.2, no methods exist for assessing the quality of the end results of a docking operation. In this case, this implies that no prior methods for computing the similarity of two boxes exist. The proposed method in this regard is to define a metric on the space spanned by all 3-D boxes. Using a metric function implies that a value of 0 indicates that the involved boxes are identical. Increasing values imply decreasing similarity. As such, the choice of a metric function means that it is really a measure of dissimilarity.

The metric is computed by identifying all the linear transformations that map the corners of one box to the corners of another box. Each linear mapping can be used to produce a set of 8 translation vectors that map between the corners of the two involved boxes. The metric is then just the minimum combined translation distance produced by such a set of translation vectors.

The reason for the existence of several linear mappings is the high degree of symmetry of a box. Or, viewed in other another perspective, the high degree of similarity of the vertices in the box graph. In total, there are 48 isomorphic ways of mapping the vertices of one box graph to another, i.e. without changing

internal connectivity of the graph. Viewed at a geometric level, this is equivalent to stating that a cube has 48 possible symmetries. Expressed in terms of matrices, the group of all 3×3 signed permutation matrices represents all the possible symmetries of a cube. A signed permutation matrix is a square matrix with two constraints on its entries:

1. Only one non-zero entry is permitted in each column and row.
2. Entries can only take the values 1 or -1.

I.e. a signed permutation matrix is any square matrix, $\boldsymbol{P}_i$, which can be created by repeatedly swapping and scaling the rows by of an identity matrix by 1 or -1. Expressing the symmetries in terms of matrices is convenient in this case, because of the decision to base the metric on linear mappings. In the current case, a 4×4 version of $\boldsymbol{P}_i$ is needed, since homogeneous coordinates are necessary to also allow for translation of the involved corners. In that case $P_i(4,4) = 1$, meaning that only the upper-left 3×3 part of $\boldsymbol{P}_i$ changes. The most convenient way of determining all the linear mappings is probably to use a standard cube as an intermediate step. This standard cube was introduced in section 3.5. The algorithm for finding a mapping from one box to another then consists of three separate transformations.

1. Map the corners of the first box to the corners of the standard cube
2. Make a signed permutation of the corners of the standard cube
3. Transform the corners of the permuted standard cube to the corners of the other box

Determining each of the involved transformations in the above is trivial. Visually, the situation is represented in Figure 4.1. In terms of matrices, the following expression yields the desired sets of translation vectors from the box β_1 to the box β_2. If $\boldsymbol{T}_{\beta_1}$ and $\boldsymbol{T}_{\beta_2}$ are 4×4 are matrices transforming the standard cube into β_1 and β_2, respectively, and $\boldsymbol{X}_{\beta_1}$ and $\boldsymbol{X}_{\beta_2}$ are 4×8 matrices containing the corners of the two boxes, then the i^{th} set of candidate translation vectors, $\boldsymbol{U}_i$ is:

$$\boldsymbol{U}_i = \boldsymbol{T}_{\beta_2} \boldsymbol{P}_i \boldsymbol{T}_{\beta_1}^{-1} \boldsymbol{X}_{\beta_1} - \boldsymbol{X}_{\beta_1} \tag{4.2}$$

Finding the distance between β_1 and β_2, $\delta(\beta_1, \beta_2)$, is then just a matter of picking the minimum combined translation produced by one of the 48 candidate sets of translation vectors, $\boldsymbol{U}_i$:

$$\delta(\beta_1, \beta_2) = \min \delta_i = \sum_{j=1}^{8} \sqrt{(\boldsymbol{U}_i{}^T \boldsymbol{U}_i)_{j,j}} \tag{4.3}$$

The design of the metric has a couple of potentially useful properties, which also makes it appealing outside the realm of 3-D boxes:

1. The matrix $\boldsymbol{T}_{\beta_2} \boldsymbol{P}_i \boldsymbol{T}_{\beta_1}^{-1}$ can be decomposed differently as a translation, rotation, and scaling matrix, which allows for diagnosis of error sources/types.

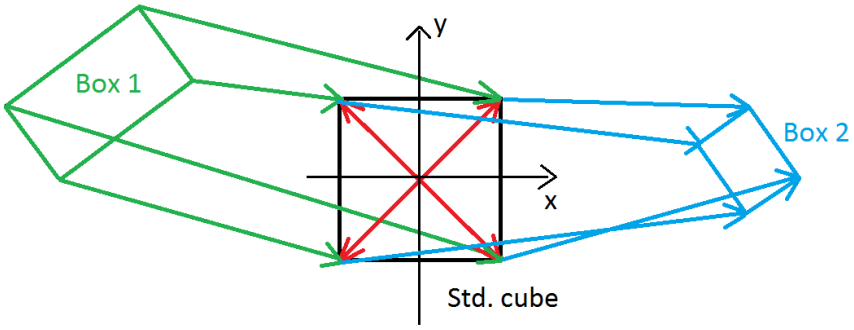

Figure 4.1: The three stage transformation from one 2-D box to another. The 3-D case is completely analogous to the 2-D case. The goal is to map the corners of Box 1 to the corners of Box 2. This is done by first mapping the corners of Box 1 to a standard cube using one transformation (green arrows). Then, some permutation of the standard cube's corners is performed (red arrows). Finally, the standard cube's corners are mapped to Box 2 (blue arrows). Only one version of the transformations from Box 1 to the cube, and from the cube to Box 2 are needed, since the internal permutation of the cube's corners handles all possible mappings from Box 1 to Box 2. The three steps of the the complete transformation are easier to determine than trying to find the complete transformation in one step.

2. The metric generalizes straightforwardly to other shapes, by defining a standardized version of that shape - just like the standard cube - and replacing the $\boldsymbol{P}_i$ with the matrices in the appropriate symmetry group for the desired type of shape. For instance, octahedrons share the same symmetry group as cubes, so in that case, the only needed adaptation is to define a standard octahedron.

3. The metric also generalizes to objects of lower or higher dimensionality, e.g. to 2-D boxes (rectangles) or 4-D boxes (tesseracts) by changing the dimensions of the involved matrices.

4. The metric is indeed a real metric, since it obeys all the requirements of metric functions, i.e. non-negativity, symmetry, identity of indiscernibles, and the triangle inequality. The easiest way to realize that this is true, is that the metric is just a minimized sum of Euclidean vector norms, which themselves are metric functions.

Experiment & Results

The performed experiment was a 2-factor within-subjects design with repeated measures for each test condition from each participant. The experimental factors were constituted by the choice of overall technique, 3C or HOC, and by the separation of DoF batches. In the HOC techniques, the DoF batches consists of

separate 3 DoF translation, rotation, and scaling operations. The design matrix of the experiment is presented in Table 4.1.

Table 4.1: The expermental design matrix of Paper A. There are two experimental factors, technique family and DoF allocation, producing a total of 8 experimental conditions. The entries in the table are read as p=position, o=orientation, s=scale. & means simultaneous defintion, $\rightarrow$ means sequential definition.

Family	DoF allocation			
	3+3+3	3+6	6+3	9
3C	p→p→p	p→p&p	p&p→p	p&p&p
HOC	p→o→s	p→o&s	p&o→s	p&o&s

The following three hypotheses were stated to set the goals of the investigation:

H1 3C techniques (with the possible exception of 9 DoF 3C) outperforms HOC techniques in terms of precision.

H2 3C techniques are faster to use than HOC.

H3 Allowing the user to focus on fewer DoF simultaneously increases the precision of the docking.

The experiment was performed using a high field-of-view (FoV), partial overlap HMD in the form of an nVisor SX111. The interaction devices were two wireless presentation mice with infrared markers attached to make them trackable by the 24-camera OptiTrack system used in the experiment. Thus, the devices offered all the functionality found on standard computer mice. The user mobility was restricted to a $2.25m$ radius by the tracking system and the HMD's power and signal cables. The experimental software was run on a Microsoft Windows PC with a custom-made OpenGL renderer. This overall setup is identical in all the experiments performed for all three contributed papers. An illustration of the used devices is shown in Figure 4.2.

In the experiment, the users were presented with a series of randomized, yellow wireframe target boxes. The task was then to use the given technique, which also came in randomized order, to dock or shape a user controlled box into the target box. For each trial, the completion time and the distance between the user and the target box were logged as response variables. 5 repetitions of each test condition was made, implying that each user went through a total of $8 \times 5 = 40$ trials.

The main results of the experiment were the following. The results are presented and elaborated in much more detail in the actual paper:

1. The 3+3+3 DoF 3C technique is significantly more precise than most of the other tested techniques.

Figure 4.2: The devices used in all experiments. *(Left)* The nVisor SX111 HMD mounted on a styrofoam head. The HMD features 1280×1024 pixels for each eye, and a total horizontal FoV of $102°$. However, the HMD only features stereoscopic overlap in the central 66% of the total FoV. *(Right)* The two wireless presentation mice used in the experiment. The mice each feature the usual 3 mouse buttons and a scroll wheel.

2. The 9 DoF 3C technique performs significantly worse than any other tested technique - as expected.

3. If the 9 DoF techniques are disregarded, then the 3C family of techniques is more precise than the HOC family of techniques

4. Controlling 3+3+3 DoF is significantly more precise than any of the other tested options. The opposite is true for simultaneous 9 DoF control.

5. Most techniques are not significantly different from each other in terms of speed. There is no systematic interpretation of the significant differences in completion times for the tested techniques.

Conclusion

The results of Paper A have demonstrated the usefulness of box shaping techniques based on the 3C representation of a box, by virtue of being more precise than comparable existing techniques. This indicates that 3C boxes and the idea of getting rid of explicit orientation specification is likely a productive approach to pursue in future studies. For this reason, 3+3+3 DoF 3C will be the technique of choice for box shaping in the other studies of this thesis.

Furthermore, the introduction of the box distance metric allows unambiguous quantification of the results of docking operations involving boxes, with the potential of generalizing to other shapes as well. Therefore, this metric should be of interest in any future docking study, where the precision of the end result is of concern.

Most of the results found in the experiment were in line with existing literature on the topic. However, the finding that 3+3+3 DoF 3C outperformed the other techniques implies that it is not always the best approach to use both hands

in a synchronous manner when shaping a volume. This result is in contradiction with current, existing results.

4.3 Paper B Summary

Introduction

Paper B is entitled *Shape Perception in 3-D Scatterplots Using Constant Visual Angle Glyphs*. This paper deals with the perception of structures, or patterns, in 3-D scatterplots explored in an immersive virtual environment. Scatterplots are one of the most common visualizations of arbitrary data. In a 3-D scatterplot, each record in a data set is represented as a 3-D object, called a *glyph*. The observed values of each record can then be mapped to various visual parameters of the glyph, such as colour, position, and shape. The idea of a scatterplot is quite old, but it is only with the aid of computers that it becomes feasible and practical to make scatterplots that feature interactive aspects.

Since 3-D scatterplots are volumetric of nature, they present a very interesting application of shaping 3-D volumes. Although 3-D scatterplots can both be explored in a 2-D desktop environment, the use of immersive technology offers a different approach, which may provide other insights through the new modes of exploration made possible in an immersive VE. As such, data exploration in immersive environments presents a possible supplement rather than a replacement for desktop data exploration.

In Paper B, there are two main contributions. One is an investigation of the volumetric density levels of glyphs needed, if the shapes of patterns in a 3-D scatterplot are to be perceptible by a human observer. The other contribution is the suggestion to use a non-standard way of rendering the glyphs, which is named constant visual angle (CVA) glyphs, which potentially allows for better perception of shapes in adverse conditions, i.e. when the desired patterns are visually occluded by clouds of clutter, or noise glyphs.

The paper is organized in the same way as Paper A. The introduction is followed by a discussion of the current state-of-the-art along with a section on CVA glyphs, which presents the theoretical background and the perceptual motivation for using CVA glyphs. The other half of the paper explains the experiment performed to investigate different glyph densities and the use of CVA glyphs.

State-of-the-Art

3-D Scatterplots

Since the production of 3-D scatterplots is a quite old discipline, many studies and tools already exist on this topic. A general history of the topic of information visualization is presented in [57]. In [67], Ware presents a very comprehensive survey of data visualization methods, including 3-D scatterplots. On page 289 of this book, Ware notes that little work has been concerned with the conditions that allow for perception of structures in 3-D scatterplots, including the effect of supporting various depth cues in the visualization. The only example given is a

study by Donoho et al [12], where the use of a spinning scatterplot is shown to improve the perception of the spatial structures inside the plot. However, the use of a spinning plot would probably not be a practical approach in an immersive VE, where the user is exploring the data from the inside.

Since then, a few notable sudies have been made. In [19], the choice of visualization parameters for 3-D glyphs is discussed in the light of perceptual psychology. Here, the notion of *perceptual grouping*, i.e. the tendency to focus on an entire pattern rather than its constituent parts, is stated as a very important aspect of human pattern perception. As will be explained, the suggested use of CVA glyphs has the potential to reinforce this ability. Perceptual grouping is also discussed in [56]. The conditions that cause perceptual grouping to occur are explained through the *gestalt laws* of perceptual psychology. These laws were originally studied in the mid 1930's [27], but remain influential today in understanding human visual perception [67]. In this paper, the most influential of the gestalt laws is the laws of *proximity* and *connectedness*, which state that objects in close proximity, and connected objects, respectively, tend to be perceived as a coherent group. These two gestalt laws are illustrated in Figure 4.3.

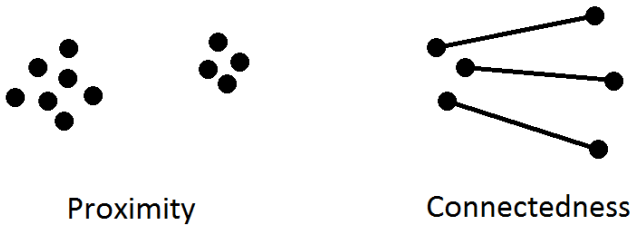

Figure 4.3: The gestalt laws of proximity and connectedness illustrated. *(Left)* The proximity law. Local proximity to other points means that most people will perceive two distinct groups of points, although the points themselves are identical. *(Right)* The connectedness law. Being visually connected is an even stronger grouping criterion than proximity. Most people perceive 3 groups of 2 points each, even though the points only form two groups by the law of proximity.

Perceptual grouping for improved pattern perception can also be reinforced without the use of 3-D glyphs. In [36], the discrete points in a data set are converted into continuous iso-density surfaces, which are then visualized. This approach also supports the notion of Paper B that 3-D glyph density is a useful and significant aspect to investigate. Further support for the significance of cloud density is presented in a 2010 study by Wang et al [66]. There, the ability of users to point out maxima in a scatterplot visualization of a 3-D temperature distribution is significantly affected by cloud density. Furthermore, the size of the glyphs is also significant. The study of Wang et al differentiates itself from the study of Paper B, by using a projection screen and only allowing the user to view the visualized data from the outside, at a distance. It is therefore interesting to see, if the results of the current study are different from those of Wang et al.

In [48], the role of illumination for shape perception in 3-D scatterplots is in-

vestigated. The idea of that paper is to use different illumation models depending on the local topology of the scatterplot. I.e. curve-shaped features will be illuminated differently from 2-D surfaces, which in turn is lit differently from volumetric features. The glyphs were visualized as points. In this case, the density of the visualization is also likely to be important, since the illumination of a 2-D surface will only look convincing if the density of the visualized point cloud is high enough. An earlier example of using illumination to increase the perception of shapes in 3-D scatterplots is found in [31].

The use of immersive VE's in the exploration of 3-D scatterplots has e.g. been studied in [40, 41]. In those works, an approach to visual data mining (VDM) using CAVE's and large panoramic 3-D stereo screens is presented. The main focus of the work is on the software architecture used, and on the idea of using dynamic visualizations, where the glyphs' 3-D positions are animated using some of the data set parameters as input.

Interactions with 3-D Scatterplots

The presence of glyphs representing undesired information is a frequent problem in 3-D scatterplots. Such glyphs can e.g. clutter the FoV, occluding the clear view of a pattern of interest. For this reason, such glyphs are referred to as *noise* glyphs. Getting rid of noise is one of the most frequent interaction goals in viewing of 3-D scatterplots.

One strategy for dealing with this issue is to introduce user-controlled clipping planes into the scene. These clipping planes deal with the noise by removing all glyphs on one side of the plane. If many clipping planes are used, it becomes possible to carve away all or most of the noise to only see the pattern. However, depending on the shape of the pattern, the planar shape of the clipping planes may be problematic, e.g. if the pattern of interest has a concave shape. The clipping plane approach is e.g. presented in [66, 76]. If a closed volume is used for clipping instead of individual planes, this approach is sometimes known as a Magic Lens [63].

Another approach to noise reduction is to change the appearance of the glyphs. This can e.g. be done by making semi-transparent glyphs, or by reducing the size of glyphs. The transparency approach is e.g. used in [60]. Reducing the size of the glyphs is only practical, if it is somehow possible to reduce the size of the noise glyphs, while keeping the size of the pattern glyphs. Otherwise, the potential gain of the operation is small, or even counterproductive, if the pattern glyphs become so small that they are difficult to see. The prerequisite of scaling the noise without scaling the pattern is a selection operation. Good and precise volumetric selections in a generic context are non-trivial to achieve, as e.g. illustrated through this thesis. Therefore, this approach is not without associated difficulties.

The third way of dealing with visual clutter, is to interactively distort the 3-D space where the glyphs are visualized, such that the noise is moved away from the pattern. Such distortions are e.g. presented in [8]. The control of the distortion can e.g. be mapped to the user's hand such that objects close to the hand are

pinched together in a smaller space, while objects further away are repulsed by the hand.

As will be explained in section 4.3, CVA glyphs represent a mixture of the latter two approaches. This is true, because CVA glyphs represents a view dependent distortion of the perceived glyph size.

There are also many interactions with 3-D scatterplots, which are not directly related to noise removal. Common interaction tasks include grand tour exploration, drill-down analysis, and brushing-and-linking. A common prerequisite of the latter two is the ability to make a volumetric selection. This highlights some of the practical applications of the topic of this thesis.

In a grand tour exploration, a large number of mappings of visual parameters are viewed in turn to reveal the projections that offer the best view of the structures in the data [1]. Such a method is both applicable to desktop and immersive contexts. Different versions of the grand tour are discussed in [40, 13].

In drill-down analysis, the exploration of the data set is done by gradually focusing on, or drilling down into, a specific region of interest. Drill-down analysis can also be combined with grand tour approaches, such that the main parameters of the visualization are gradually identified to arrive at the best possible view of the pattern of interest. In [13], a strategy involving both a modified grand tour and drill-down analysis is presented. An essential prerequisite of drill-down analysis is to be able to specify the 3-D region to drill into.

In brushing-and-linking, the goal is to highlight an interesting pattern of glyphs (called brushing) to see how it behaves using different projections of the data set (called linking), e.g. to find otherwise hard-to-detect relationships between a large number of observed variables. The brushing-and-linking technique was introduced by Buja et al in 1991 [7]. Once again, volumetric selection is a necessary prerequisite, which allows for the actual brushing to occur.

Contributions

CVA Glyphs

The definition of a CVA glyphs is a glyph which is rendered such that its projected size does not change as the viewpoint moves closer or further away. Under normal circumstances, this implies two things:

1. CVA glyphs cover a constant amount of pixels in screen-space.
2. CVA glyphs cover an *approximately* constant part of the viewer's FoV[1]. Hence, the name constant visual angle glyphs.

Although the visual angle covered by the CVA glyphs is only approximately constant, the CVA name is chosen, because it better describes the perceptual consequences of rendering glyphs in this way.

[1]This is only true, when the glyphs are not rendered using a very large perspective FoV, in which case CVA glyphs rendered at the edge of the FoV would cover siginificantly smaller visual angles than those rendered in the centre of the FoV.

The idea of rendering objects with these properties is not new at all. However, the potential perceptual utility of doing so in the context of exploration of 3-D scatterplots has not been noted or investigated before. The most prominent example of a rendering method, which would always produce CVA glyphs, is the use of orthographic projection. However, orthographic projection is not very well suited for immersive viewing, since it provides a very unnatural viewing experience. Furthermore, the real power of CVA glyphs only shows, when the glyphs are rendered in a regular perspective context.

The point of using CVA glyphs is that they have the potential to reduce the effects of noise in a visualization without requiring the user to employ any special tools, such as hand-controlled distortions or clipping planes. The only required interaction is the ability of the user to navigate to a place, where the pattern is visible without much noise getting in the way. This is true, because of the perceptual consequences of using CVA glyphs:

With glyphs rendered using regular perspective projection, the visual angle of glyphs increases as the viewpoint moves closer, and vice versa when the viewpoint moves away. This produces the well-known effect that nearby objects cover a larger amount of the FoV than those further away. Since this is also the way that the human visual system works in the real-world, this effect, called perspective foreshortening, is very important in the assessment of the distance to objects. I.e. perspective foreshortening is a significant *depth cue*. Even though the visual angle of an object changes, the usual human interpretation of this is that the object has retained its physical size, but changed its distance to the viewer. This is known as the size-distance invariance hypothesis [4]. However, since the visual angle of a CVA glyph does not change when moving relative to it, the basic assumptions of this hypothesis are violated. Instead, the most likely interpretation of seeing a CVA glyph in a head-tracked VE will be that the physical size of the glyph is changing to accomodate that its visual angle is not changing, since the viewer intrinsically knows the amount of self-motion relative to the glyph. This is illustrated in Figure 4.4.

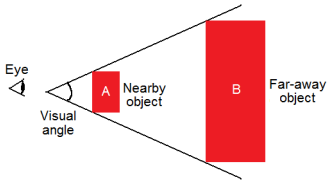

Figure 4.4: The size-distance invariance hypothesis in relation to CVA glyphs illustrated. Although the two objects, A and B, cover the same visual angle of the viewer, they are perceived to be of different size and at different depths. If the viewer is able to move closer to the objects, and the visual angle does not change (the case with CVA glyphs), then the only natural inference is that the objects have become smaller. The opposite is true when backing away. Therefore, CVA glyphs violate the size-distance invariance hypothesis.

This unusual phenomenon does not rule out the use of any other depth cues,

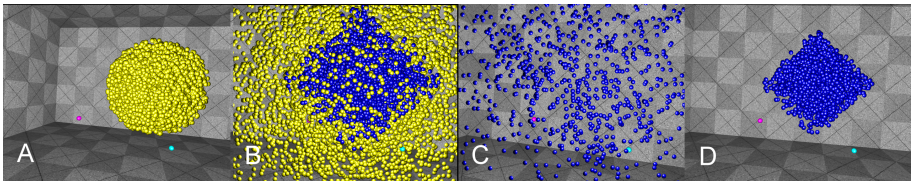

Figure 4.5: A CVA glyph cloud with the noise-spreading and pattern-solidification properties illustrated. Yellow glyphs are noise, blue ones are pattern. Note that all glyphs are the same size in pixels across all 4 images. (A) The cloud is viewed at such a large distance that none of the blue pattern is revealed. Instead, the noise is solid. (B) Moving closer to the cloud reveals the blue pattern, relatively unobscured by the noise. This situation also represents a sweet spot, where noise-spreading and pattern-solidification are both helping. (C) Standing inside a pattern causes it to spread out and become imperceptible. (D) Moving away from the pattern causes it to solidify into a recognizable shape.

such as linear perspective, stereo, shading, cast shadows, and occlusion. Most importantly, it is still possible to let the positions of CVA glyphs be subjected to regular perspective projection, even if the rendering of the glyph itself on-screen follows the unusual properties of CVA glyphs. This means that the perceived spatial relationships between CVA glyphs is intact, even if the perception of the size of the individual glyphs is subject to a view-dependent distortion. Thus, the structures and patterns in a 3-D scatterplot of CVA glyphs are still intact. Furthermore, If CVA glyphs are combined with linear perspective, two potentially very useful properties are introduced:

1. **Noise-spreading:** Because of linear perspective objects move outwards in the field-of-view as the viewpoint moves closer. At the same time, the CVA feature ensures that the objects do not increase their coverage of the FoV. The combined effect is therefore that a user can thin out any noise to see what is found behind, simply by moving closer to the noise.

2. **Pattern-solidification:** This is opposite of noise-spreading. If a viewer is standing inside a cloud of interesting glyphs, it is possible to solidify that cloud by backing away from it. This is true, because linear perspective ensures that the screen positions of objects condense as you move away. At the same time, the CVA feature implies that the glyphs do not decrease in visual angle. Eventually, the CVA glyphs will begin to touch and overlap on-screen, which solidifies the pattern.

In many cases, it is possible to adjust the viewpoint such that both of the above properties are used. I.e. such that the noise is removed, while the pattern is solidified, representing a kind of sweet-spot for viewing that pattern. This also implies that the CVA feature has reinforced the perceptual grouping, while getting rid of any visual occlusion. Both of the properties are illustrated in Figure 4.5.

The actual implementation of CVA glyphs is most easily and efficiently achieved using point sprites. That is, images of the glyphs, which are rendered in screen

space after any perspective projection is performed. This approach also allows the glyphs to be shaded to appear as real 3-D objects instead of flat 2-D images. The main constraint, however, is that spheres are the only practical glyph shapes to use, if 3-D lighting and shading is desired.

Experimental Results

The goal of the experiment was to investigate the effects of three factors: Glyph density, the presence/absence of noise, and the use CVA glyphs compared to glyphs rendered with regular perspective projection. The noise and CVA factors were both binary factors, i.e. on or off, while the density was tested through five chosen levels. This produced a total of $2 \times 2 \times 5 = 20$ unique conditions. The five density levels were chosen such that a clear difference was visible, even without the HMD, and such that the pattern shape was indiscernible at the lowest level, and clearly visible at the highest level. Thus, testing this range of densities should reveal where any threshold levels might be. The specific levels were: 1,000, 5,000, 10,000, 50,000, and 100,000 glyphs/m^3. The factors and levels are presented in Table 4.2.

Table 4.2: A table of all test conditions of Paper B. The numerical entries in the table are the unique ID's assigned to each condition. E.g. condition 4 has a density of 5,000 glyphs/m^3 with CVA glyphs enabled and no noise.

Without noise					
CVA	Density [glyphs/m^3]				
	1,000	5,000	10,000	50,000	100,000
Yes	0	4	8	12	16
No	1	5	9	13	17

With noise					
CVA	Density [glyphs/m^3]				
	1,000	5,000	10,000	50,000	100,000
Yes	2	6	10	14	18
No	3	7	11	15	19

The glyph size was either fixed to a diameter of 25 pixels in the case of CVA, or to a radius of 1 cm in the case of regular perspective glyphs. The use of CVA glyphs, which violate the size-distance invariance hypothesis, makes it impossible to design the experiment such that the different types of glyphs always appear to have the same size in both cases: With CVA, the glyphs will appear to change their perceived visual size, and appear small when the user is close. Without CVA, the visual angle of a glyph grows as the user moves close, eventually covering the entire FoV of the user. This fact is unavoidable and expected, no matter which absolute size is chosen for the non-CVA glyphs. Using a CVA size of 25

pixels implies that the physical, perceived size of the glyph is approximately 1.44 cm when viewed at a distance of 1 m. Conversely, the viewing distance, which produces a physical radius of 1 cm is around 69 cm. I.e. at the chosen glyph sizes, the perspective glyphs cover more screen space than the CVA glyphs if the user is closer than 69 cm. Further away, the CVA glyphs cover more screen space. It is therefore a reasonable conclusion that the size difference does not unreasonably favor any of the glyph types at realistic viewing distances. The formula used when calculating the relation between a screen-space size and a physical radius can be derived from knowing the parameters of the projection and viewport transformations as given in Equation 4.4:

$$r = \frac{s}{2} \cdot \frac{p_z}{n} \cdot \frac{t-b}{h} \tag{4.4}$$

In Equation 4.4, r is the physical radius that would currently result from a sphere of pixel diameter s, if the sphere is viewed at a distance of p_z with a vertical screen resolution of h pixels, using a perspective projection with top, bottom and near clipping plane parameters, t, b, and n. Isolating s instead of r, inverts the formula such that the on-screen size of a glyph is computed at a given radius. This version of the formula represents the normal mode of operation of a rendering pipeline, where a physical radius must be converted into a size measured in pixels.

Five different hypotheses were stated to set the frame of the investigation:

H1 CVA glyphs will provide better pattern perception and pattern reproduction performance than regular glyphs in noisy environments.

H2 CVA glyphs will not pose a problem to the perception and interactions of the users compared to regular glyphs.

H3 There exists a lower threshold below which pattern perception becomes difficult and pattern selection performance decreases.

H4 There exists an upper threshold above which pattern perception becomes easy and pattern selection performance increases.

H5 There is statistically significant interaction between density and the use of CVA glyphs.

The experimental equipment used was the same as that used for the experiment of Paper A. In the experiment, the participants were presented with a series of randomized clouds of blue pattern glyphs, which in the case of noise were completely surronded by a yellow noise cloud. The pattern cloud was always box-shaped, while the noise cloud was spherical to avoid revealing any spatial features of the pattern. The goal of the participants was to inspect the cloud and recreate the minimum bounding box of the blue pattern cloud. This choice allowed the application of the 3+3+3 DoF 3C technique and the box similarity measure introduced in Paper A. Furthermore, the participants would be asked to both quantify the difficulty of perceiving the minimum bounding box

of the pattern, and the difficulty of reproducing this box. This choice allowed the use of both objective (box similarities and completion times) and subjective (difficulty ratiings) measurements to be made. After a short training session at the beginning of the experiment, the order of experimetal conditions was randomized. Each participant did two repetitions of each trial.

The main results of the experiment were the following (see the actual paper for more details):

1. CVA glyphs outperform perspective glyphs in noisy contexts, supporting the statement of the first hypothesis.

2. Correctly perceiving pattern shapes at the lowest tested density level is significantly more difficult than any of the higher density levels.

3. Completion times are only significantly affected by the presence of noise.

4. The most difficult condition tested is the case with a high density of both noise and pattern.

5. The lower density threshold below which pattern shape identification becomes significantly harder lies somewhere in the 1,000 to 5,000 glyphs/m^3 range.

6. The upper threshold above which pattern shape perception becomes significantly easier is somewhere in the 10,000 to 50,000 glyphs/m^3 range.

7. The use of CVA glyphs is preferable in noisy or high density contexts, while the use of regular perspective glyphs is preferable in situations with low density or without noise.

Conclusions

The contributions of Paper B were both the introduction and analysis of CVA glyphs, the evaluation of their usefulness, and the investigation of density levels in immersive visualizations of 3-D scatterplots. The analysis of CVA glyphs had a particular focus on the perceptual implications of rendering objects, which are of constant size in screen-space.

The main conclusions of Paper B are twofold. In relation to CVA glyphs, the introduction of such glyphs can be beneficial in immersive exploration of 3-D scatterplots, when the viewing conditions are otherwise adverse due to the presence of clutter. Viewed in a broader context, the results regarding CVA glyphs may also inspire future techniques, where the removal or modification of natural depth cues can be used in a beneficial way. In relation to density levels, the discovery the location of possible threshold ranges, where the perception of patterns is improved or degraded can be used in several ways. If used at an early stage of an experiment, it may be used to guide researchers in the number of samples needed, if subsequent visual analysis through 3-D scatterplots is desired. The result may also be used to enhance pattern perception during an exploration, by

lowering the glyph density outside of a pattern, while increasing it inside that pattern.

Both the 3C box shaping techniques and the box similarity measure were used in the study. Therefore, the experiment also presented an opportunity to put the ideas presented in Paper A to practical use outside of the docking experiment tested there. In this case, the context was to be able to precisely, objectively, and quantitatively assess the degree of perception of a box-shaped pattern. As the following section will elaborate in much greater detail, the third paper of this thesis will also apply a 3C technique in yet another context. The third paper will also make use of both CVA glyphs and the knowledge gained about density levels through the experiments of Paper B.

4.4 Paper C Summary

Introduction

Paper C is entitled *Brush, Lasso, or Magic Wand? Picking the Right Tool for Large-Scale Multiple Object Selection Tasks*. Its aim is to investigate the differences between three techniques for doing multiple object selection (MOS) in immersive contexts. MOS refers to any task with the goal of selecting more than a single object. Such tasks are frequently seen in desktop environments, where it is common to select a large number of objects to be the targets of the same operation(s). Since there is nothing 2-D desktop specific about this idea, it is equally relevant in immersive 3-D contexts. Examples of immersive application areas include data mining, medical visualizations, and interactive model editing. However, the applicable techniques for solving MOS tasks are different on a 2-D desktop and in immersive 3-D environments. This makes 3-D MOS an interesting topic to investigate. Since one of the most common ways of selecting items in 3-D space is to use a selection volume which indicates the object(s) of interest, MOS is a prime example of the utility of efficient shaping of 3-D volumes.

One obvious way of solving MOS tasks is to use a single-object selection (SOS) technique in a sequential manner, such that each single-object selection adds an object to the selection set. However, as the number of selection targets, henceforth called the *scale* of the MOS task, increases, so does the impracticality of the sequential SOS approach. This implies that there not only is a distinction to make between MOS and SOS, but also between MOS on a large scale (LS-MOS), and MOS on a small scale (SS-MOS). Thus, the techniques suitable for LS-MOS are likely to be very different from those applicable to SOS and SS-MOS. In Paper C, the focus is therefore on LS-MOS tasks, since such tasks present challenges which are very different from those of the well-established area of SOS.

One of the tested methods, called a magic wand, is new in an immersive 3-D context. The main focus of the performed experiment is to see how the magic wand technique fares against existing techniques, and to investigate how the spatial layout of the selection targets changes the performance of the techniques. Paper C therefore has the potential to aid designers of immersive applications that include MOS tasks in their lists of requirements.

State-of-the-Art

The term *selection* in current literature on interaction techniques for VE's, almost exclusively refers to the task of selecting a single object in the virtual world. Very few studies of 3-D MOS exist, and existing literature on selection often fails to mention or acknowledge the possibility wanting to select multiple objects. Even standard textbooks on interaction in 3-D environments fail to mention MOS in otherwise quite general taxonomies of interaction tasks [6]. This is quite surprising, since MOS is clearly a type of selection task. This implies that selection is a more general task category than the impression left by most existing works on the topic. For this reason, selection taxonomies ought to include both SOS and MOS.

If 3-D MOS in general seems somewhat neglected in current studies of the topic, this is even more true of LS-MOS, where the scale of the task is well into the hundreds. I.e. several hundred or more objects need to be selected.

3-D MOS

Most of the existing literature which specifically deals with 3-D MOS is identical to the literature dealing with the shaping of 3-D volumes. This is the case, because the goal of shaping a 3-D volume in existing studies has always been to make a selection of multiple objects. Examples of this are e.g. found in [61, 32], where 3-D boxes are used for volumetric selection.

Viewed at a very general level, which does not only apply to 3-D contexts, MOS techniques can be split into three overall categories. Two of these categories, called *brushes* and *lassos*, were introduced by Wills in [70]. A brush is a technique, which uses a *persistent* selection volume, whereas a lasso is a *non-persistent* selection volume. Being persistent, implies that the volume is often attached to the user's hand, which allows the user to sweep the volume in space to make a selection which is more complex than the shape of the brush volume. The lasso is non-persistent, because it is defined by the user in-situ to make a selection of whatever falls inside the volume. Once a lasso is defined, it disappears, and the user has to define a new lasso from scratch in another location to expand the selection.

The most well-known example of lassos from 2-D contexts is probably the rubber band selection rectangles used when picking multiple icons on a desktop. The studies of Ulinski et al [60, 59, 61, 58] employ lasso-type selection techniques, in the form of non-persistent 9 DoF boxes. These studies also represent some of the few ones which may be relevant for LS-MOS. The other existing examples of LS-MOS mainly come from medical applications. E.g. MRI scan visualizations are typically composed from thousands of small voxels, of which only a small subset is of interest to the doctor inspecting the MRI scan. This is a compelling case where efficient and specialized 3-D LS-MOS is essential for precise annotation and analysis of the data. In [16], an interactive desktop tool for this purpose is presented. This technique is based on editing a 3-D mesh at the vertex level to make it fit specific anatomical structures in the scanned data. Nakao et al [42] also suggest using deformable meshes to cut out volumes of interest from MRI data.

We have chosen not to implement any of these methods, as they are specifically designed for inspection of MRI voxel type data in desktop environments.

The best example of a brush from the 2-D desktop world is probably the brush tools used in drawing and painting applications, where the painting of pixels is equivalent to selecting and modifying multiple pixels. In 3-D, the studies by Lucas et al [33, 32] amounts to brush type selection, since a persistent selection box is used. The studies by Lucas et al are all done on a relatively small scale, where the maximum number of tested selection targets was 64. This maximum is probably heavily affected by the fact that the studies also included serial SOS as one of the tested techniques. Thus, Paper C's aim of hundreds or a few thousand objects is well above the number tested there. The studies by Lucas et al also present a design space for 3-D MOS techniques, which includes 6 design parameters:

1. Spatial context. Whether the technique is performed in the image plane or directly in 3-D space.

2. Concurrency. Whether the technique achieves MOS in a parallel or a serial fashion. The latter implies using a SOS technique for MOS.

3. Indication object persistence. Essentially whether the selection volume is a lasso or a brush.

4. Indication object dimensionality. Whether the selection volume is a 0-D, 1-D, 2-D, or 3-D shape.

5. Indication object creation. Whether the selection volume is created directly in 3-D space, or indirectly, e.g. by picking it from a list.

6. Indication object manipulation. Whether the selection volume is manipulated directly or indirectly.

This design space is quite comprehensive, but fails to account for techniques which do not rely on using an explicit indication object, or selection volume, to make the selection. Such techniques represent the third category of 3-D MOS techniques, which should also be accounted for in the complete design space of MOS.

The third MOS category, which is neither part of Wills' design space presented in [70], nor of Lucas' design space [32], accounts for the possibility of partly automatizing the MOS procedure. This implies that the user initiates the selection by pointing out one or more objects of interest, after which an algorithm expands this initial selection, or *seed*, to a larger set of similar objects. This approach to MOS has not been very widely studied, and the only techniques currently in this category are ones which require a scene graph or a similar hierarchical structure to expand the initial selection. Examples of expanding a selection through a hierarchical data structure are selection of all objects standing on the floor of a room by selecting the floor, or selection of all the objects on a table, by pointing out the table. Such techniques are e.g. presented in [53, 24, 71]. However, in the typical use of LS-MOS, i.e. data visualizations, no such hierarchy exists, meaning

that techniques requiring a hierarchy are not possible. Therefore, Paper C suggests and evaluates a new technique in 3-D contexts, the *magic wand*, which is introduced in more detail in section 4.4.

In [50], Steed presents an attempt at making a general mathematical model of all selection techniques. The proposed model is based on using time-varying scalar fields to represent the motion of a selection volume through a 3-D space. However, this model does not immediately capture the proposed magic wand technique, since the magic wand does not define any explicit selection volume.

3-D SOS

Techniques for selection of single objects are also of interest in a MOS context. First of all, any SOS technique used in a serial way is equivalent to a MOS technique. Such techniques, however, become increasingly impractical to use as the MOS task scale grows, as e.g. demonstrated by Lucas et al [32]. This link between SOS and MOS is therefore not so relevant on large scales, which is the focus of this paper.

There is a second link between SOS and MOS, however. This link is formed by the fact that selection of a single object very often involves a disambiguation step/mechanism. Disambiguation is necessary because SOS techniques are very often designed such that it is both possible and highly likely that more objects are possible candidates for the intended target. A prime example of this is the ray casting technique [37]. There, the ray will frequently intersect more than one object, which calls for disambiguation to pick just one of the candidates. Examples of such disambiguation mechanisms for ray casting are e.g. presented in [28, 11], and for cone casting in [51, 50].

The applicability of this link between SOS and MOS in an LS-MOS context is highly dependent on the nature of the SOS technique. For instance, a ray only forms a very small and narrow selection volume, implying that the amount of intersected candidates is likely low. This means that the use of a selection ray without disambiguation is probably only feasible in very small scale MOS tasks. The use of a cone used as a flashlight instead of a ray fixes this problem somewhat.

Contributions

The Magic Wand Technique

The first contribution of Paper C is the introduction of the magic wand technique to 3-D contexts. Magic wand selection tools are well-known in desktop contexts, especially in image processing and painting applications, where the wand is used to select a large region of similar pixels. In 2-D, this is done by using the wand on an initial, representative pixel, after which an algorithm is used to automatically expand the selection to a larger region of similar pixels which are connected to the initial seed. Since the magic wand is already a well-established metaphor for this kind of tool in image processing, this name is also chosen for its 3-D counterpart.

In image processing, the region growing is somewhat simplified by the fact that all pixels are arranged in a rectangular lattice with well-defined neighbours

of all pixels. This would also be true in a 3-D visualization based on voxels, e.g. from an MRI scan. However, for 3-D scatterplots like those studied in Paper B, the case is more difficult, because ground truth connectivity between the data samples is *not* given a priori. This means that the 3-D magic wand technique must include a mechanism for producing this connectivity information.

The desired outcome of using the magic wand is essentially the same as the output of a clustering algorithm, with the constraint that only one cluster containing the seed is desired. The proposed algorithm to base the magic wand on, therefore comes from the class of clustering algorithms. Since the magic wand is an interaction technique to be used by humans, the results of using the magic wand should preferably correspond to the results that would be expected by a human. Furthermore, the magic wand should be usable in a range of scenarios to make it a generally applicable tool. As such, the clustering algorithm should not depend on the assumption of a specific distribution of the selection targets. To ensure this, the gestalt laws of perception [27, 67] are once again useful. The final choice is to design the magic wand such that its function corresponds to the gestalt law of proximity, such that the local proximity to other objects in the cluster is the deciding factor.

Such an algorithm is e.g. presented as the initial grouping step of a more sophisticated clustering algorithm in [18]. In this algorithm, data samples are included in the cluster by the criterion that the closest object to *any* member of the current cluster is added next. As such it represents a kind of single-linkage clustering [49]. This means that clusters will grow organically, based on local conditions, rather than global conditions. I.e. samples far from the seed can be added to a cluster, if there is just a single cluster member which is close enough. Pictorially speaking, this means that any member of a cluster can be reached by a series of sufficiently small jumps through the cluster from object to object, much like crossing a wide stream using stepping stones. An example of this is illustrated in Figure 4.6.

In the case where the goal is only to find a single cluster and leave the rest of the samples unclustered[2], there must be some criterion to stop the clustering once there are no more similar, connected objects to add. To stay in line with the gestalt law of proximity, this stopping criterion should be based on local conditions rather than global conditions. Specifically, an upper limit on the distance allowed between an object and its nearest neighbour solves the problem. Without this upper limit, the clustering would eventually include all available objects. Raising this limit expands the cluster, while lowering it reduces the cluster. In an interactive context, this both provides a useful and a meaningful parameter to control.

There are many other possible clustering control parameters that could be added to the magic wand technique, however the upper limit on internal distance is the only one implemented and evaluated in Paper C. Possible future parameters include putting limits on the number of jumps from the seed, on the

[2]In other words, put all other samples into one large cluster irrespective of the internal similarity of this cluster.

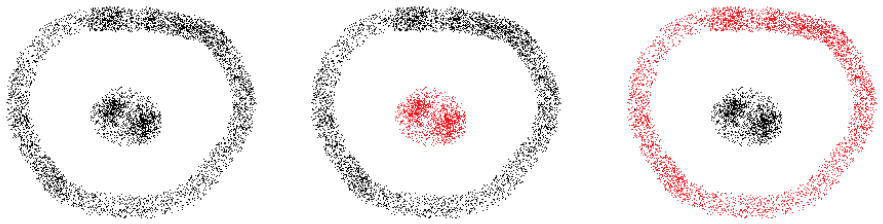

Figure 4.6: A 2-D illustration of magic wand cluster growth. (*Left*) The unclustered data. The gestalt law of proximity clearly implies that a human would perceive 2 distinct clusters: An inner, circular cluster, and an outer, ring-shaped cluster. (*Middle*) Selecting a seed from the inner cluster expands the selection to all points of that cluster by the local proximity criterion. This cluster could also be easily found by a global criterion, such as Euclidean distance to the seed. (*Right*) The seed is selected from somewhere in the outer cluster. The local proximity criterion ensures that the clustering only includes samples from the outer ring. Using a global criterion like Euclidean distance from the seed would erroneously include the inner cluster in the selection.

accumulated distance from the seed, or on the simple Euclidean distance from the seed.

Experiment & Results

The experiment is based on the idea of evaluating the magic wand against a brush and a lasso in a number of scenarios representing different spatial layouts of the selection targets, in this case 3-D glyphs. This means that one factor of the study is constituted by the three tested MOS techniques, while the other is made up of five different geometric scenarios.

The brush technique is designed as two semi-transparent sphere attached to hands of the user. The radii of the spherical brushes are independently adjustable using the scroll wheels of the mice. Semi-transparency is used to allow for better perception of the objects inside the brush. This is in line with the results of a 1994 paper by Zhai [73], which introduces a semi-transparent selection box named the silk cursor. The brushing is activated by clicking and holding a button on the interaction devices. Since the user can adjust the position and radius of the spheres, this adds up to a total of 4 DoF for the designed brush. Had the brushes been non-spherical, such that orientation was also a DoF, this would imply up to 3 additional DoF. The magic wand is also a 4 DoF tool, since the user can control a 3-D position, and a threshold value.

The lasso technique is implemented using the 3+3+3 DoF 3C box shaping technique introduced in Paper A. The reason for choosing a box-shaped lasso is mainly that it seems to be the natural extension of the 2-D desktop rubber band selection rectangles. However, the choice of a box also implies that the chosen lasso has 9 DoF compared to the 4 DoF of the spherical brush and the magic

wand. This implies that the amount of cognitive and practical work going into a single lasso is somewhat larger than that of the brush. This disadvantage can possibly be offset, if a box is a better match for the geometric scenario, e.g. if the glyphs are organized in a box-shaped lattice, or otherwise in an approximate box shape.

The geometric scenarios are designed such that they represent various aspects visualizations of real-world data. Furthermore, they are designed such that they potentially highlight the strengths and weaknesses of each technique. I.e. some combinations of techniques and scenarios should represent very simple tasks, whereas other combinations are almost impossible. A scenario consists of a cloud of blue selection targets and cloud of yellow, undesired glyphs, called noise. The target and noise clouds are always non-overlapping to allow for perfect selection results, at least in principle. This choice means that the upper bound on performance is known a priori. The five chosen scenarios are (see Figure 4.7 for an illustration of the scenarios):

Separated Clouds (SC) The separated clouds scenario presents the user with two completely separated, spherical clouds. Because of the separation and convexity of the clouds, this scenario is expected to be easy, no matter which technique is used.

Adjacent Clouds (AC) The adjacent clouds scenario places two box shaped clouds immediately next to each other, only separated by a very small gap. This means that one face of the bounding boxes of the two clouds almost touch. This scenario should be easy to solve with the box-shaped lasso, but represents a bad scenario for the magic wand, because local proximity is no longer enough to separate the clouds. The brush is expected to be somewhere in-between.

Entangled Clouds (EC) The entangled clouds are presented as two toroidal clouds, which are entangled such that they are well-separated everywhere. Specifically, one torus runs straight through the geometric centre of the other. This scenario is the perfect case for the magic wand's local proximity criterion, whereas the box-shaped lasso is expected to be a poor fit for the task. Once again the brush is somewhere in-between the other two.

Embedded Nucleus (EN) In the embedded nucleus scenario, a dense, box shaped cloud of selection targets is surrounded by a sparse cloud of noise. Thus, the lasso is a perfect fit, while the brush and magic wand are more dubious. However, the magic wand probably has a chance of achieving good results in a short amount of time, due to the higher density of the selection targets.

Uniform Embedding (UE) The uniform embedding is identical to the embedded nucleus, except that both clouds are equally dense. This means that the scenario represents a worst-case situation for all of the tested techniques.

Due to the findings of Paper B, the density of the clouds is 50,000 glyphs/m^3 in all cases, except for the noise cloud of the EN scenario. Furthermore, the glyphs

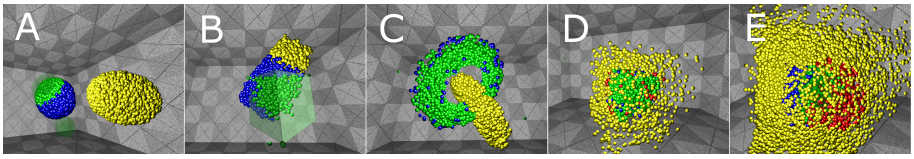

Figure 4.7: Examples of all scenarios and techniques used in the MOS experiment. The blue/green glyphs represent the selection targets. The yellow/red ones are undesired noise glyphs. (*A*) The separated clouds scenario with the green glyphs selected using a brush. (*B*) The adjacent clouds scenario with a selection box lasso. (*C*) The entangled clouds scenario, where the magic wand has been used. The remaining blue glyphs can be selected by adjusting the local proximity threshold. (*D*) The embedded nucleus scenario. The selection has been made using the magic wand. A few false positives are showing as red glyphs. (*E*) The uniform embedding scenario. The glyphs have been selected using a spherical brush.

Table 4.3: All combinations of the two factors of Paper C's experiment. The numbers in the table will be used to refer to the specific combinations of the two factors in the analysis of the results. The (+), (.) and (-) labels indicate if the technique is expected to perform well (+), average (.), or badly (-) in the given scenario.

Technique	Scenario				
	SC	AC	EC	EN	UE
Lasso	0 (+)	1 (+)	2 (-)	3 (+)	4 (-)
Brush	5 (+)	6 (.)	7 (+)	8 (.)	9 (-)
Wand	10 (+)	11 (-)	12 (+)	13 (+)	14 (-)

are rendered as CVA glyphs to make the UE scenario feasible. I.e. the UE scenario features targets completely embedded in dense noise. This makes the targets completely occluded, unless some noise-reduction technique like CVA glyphs is employed.

Examples of all five scenarios and the MOS techniques in action are shown in Figure 4.7. The choice of using three MOS techniques and five geometric scenarios implies that there are a total of $3 \times 5 = 15$ unique conditions in the experiment. These conditions are presented in Table 4.3.

The experiment itself was designed as a randomized within-subjects experiment, just like the experiments of the first two papers. The participants would make 3 repetitions of each test condition. The response variables consisted of both subjective and objective measurements. The selection quality was e.g. assessed through the sensitivity and specificity of the selection. A count of the number of operations used in each selection was logged as an objective measurement of difficulty. As with the previous experiments, the completion times of

all trials were also logged. The difficulty of making a good selection was subjectively judged by the participants on a 1-10 scale. A few secondary response variables were also computed from the primary ones, to be able to better analyze the differences between the techniques and scenarios:

The sensitivity/operation count ratio indicates how much of the correct parts of the final result was achieved by each selection operation.

The operation count/completion time ratio indicates how fast each selection operation was.

The sensitivity/completion time ratio indicates how fast the correct parts of the solution were achieved.

The equipment used in the experiment was the same as the previous experiments, except for an upgrade of the PC running the experimental software. The hypotheses of the experiment were the following:

H1 The combination of MOS tool and geometric scenario significantly affects the selection performance.

H2 The box lasso is better than the other techniques in the adjacent clouds (AC) scenario.

H3 The magic wand is better than the other techniques in the entangled clouds (EC) and the embedded nucleus (EN) scenarios.

H4 Overall, the magic wand is faster to use than the other techniques.

H5 Overall, the brush is easier to use than the other techniques.

There were 18 participants in the study. The results were mainly in support of the hypotheses. However, the box lasso consistently performed worse than expected, even in the scenarios specifically designed to be easy for a box lasso to complete. Put briefly, the main results are (see the actual paper for more details):

1. There is a very significant interaction between the choice of MOS technique and the geometric scenario. This both serves as strong support for the general idea of the experiment, and in particular for the first hypothesis. The result means that it is important to pick the right tool for a MOS task, depending on the geometric scenario.

2. The box lasso was better than the magic wand in the AC scenario, as expected. However, it did not outperform the brush, unless the sensitivity/operation count ratio is considered. This means that the box lasso is only better in the AC scenario, if the goal is to solve the task well in a few operations. Thus, H2 does not get much support by the data. This means that the shape of a MOS selection volume is less important than other factors in judging how fit a MOS tool is for a given task.

3. The magic wand does outperform the other techniques in the EN and EC scenarios, but mainly with respect to speed. I.e. it is possible to achieve equally good results with all three techniques in the EN and EC scenarios, but the magic wand will achieve the results significantly faster.

4. The brush is subjectively judged to be easier to use than the other two tools. However, the number of operations used with the brush is significantly higher than the other two techniques. Thus, there seems to be no connection between the number of times that a tool is used, and the perceived difficulty of using the tool. This implies that a low DoF tool used serially is likely to be preferred over using a precise high DoF tool a few times - at least for non-expert users.

5. Participants seem to have an asymmetrical perception of errors induced by low sensitivity versus low specificity. It seems that the participants in difficult cases prefer getting a low sensitivity, if this implies getting a high specificity. I.e. false positives are judged to be worse than false negatives. This observation is also statistically verifiable in the experimental data.

Conclusion

There are several conclusions to the work presented in Paper C. First of all, the introduction of the magic wand technique offers a new way of performing MOS on a large scale. The magic wand idea introduces a cost-benefit ratio to be considered when deciding whether or not to use the wand. The cost can be viewed as the lack of control induced by the automatic clustering performed by the magic wand, and the benefit is the amount of work it saved by the user. At small MOS scales, there will be little or no benefit of using a magic wand, while the cost is still the same. Thus, the use of serial SOS techniques at small scales is probably preferable, since it features the advantage of not risking false positives selected by an algorithm. At large scales, the cost-benefit ratio is vastly in favour of the magic wand, beacuse of the large work reduction on behalf of the user. Furthermore, a few false positives are probably less important in LS-MOS contexts, since these false positives represent a much smaller fraction of all the selected objects than in SS-MOS cases. The magic wand can probably be improved quite a lot by including additional controllable parameters, and by using more data dimensions in the distance measure used for the local proximity criterion. E.g. with real-world data, differently coloured glyphs would likely imply that some attribute of the involved samples has different values. If this attribute is included distance measurements, then even spatially close glyphs would be easily separable.

The results of the experiment allows the characterization of the good and bad features of each of the tested techniques:

1. The spherical brush is very versatile, yielding good selection performance in most scenarios. However, it is often slow to use and requires many strokes and size adjustments to reach the goal.

2. The box lasso is good at selecting many objects with each operation. However, it is slow and judged to be difficult to use, even when the selection targets are perfectly arranged for a box-shaped selection volume.

3. The magic wand is faster than any of the other techniques. It reaches high quality selection results in a short amount of time, if there is good separation between selection targets and noise, or if the noise is less dense than the target. On the other hand, the magic wand performs very badly, when the opposite is true of the geometric scenario.

The experiment has demonstrated that it is important to carefully consider the choice of MOS tool depending on the geometric scenario. However, quite unexpectedly, a good MOS tool for a task has been found to not necessarily be one, which matches the overall shape of the cloud of selection targets. Instead, it seems that the repeated use of a simple tool is preferable to a few uses of a complex tool. One interesting point to make here is how to measure MOS tool complexity. The best guess based on the studies made through this thesis is that the DoF of a tool are important. It therefore seems an interesting avenue of further investigation to perform an experiment, where the number of MOS tool DoF is one of the factors. This could also help uncover if there really is any difference between brushes, lassos, and magic wands, or if the found differences are mainly due to a more fundamental parameter such as tool DoF.

5 Conclusion

This thesis has presented three papers. Each of these papers brings some new ideas and techniques to the field of 3-D user interaction, specifically within the topic of shaping and perceiving 3-D volumes. Furthermore, each paper also presents the results of an experiment, which evaluates the potential usefulness of the suggested techniques.

The first paper, Paper A, presents the novel idea of letting a user define a 3-D box by indicating where just three of its corners are to be placed, giving rise to the name 3C for such techniques. The mathematical considerations that lays the foundations of that idea were also elaborated in more detail in chapter 3. The main motivation for designing a box shaping technique in this way is the difference in difficulty between indicating positions and orientations. This difference is well-known and well-established in all current literature on the topic. By shaping a box through the direct placement of box corners, the difficult task of precisely indicating an orientation is transformed into the simpler task of precisely indicating positions. It is very likely that this idea can also be generalized to other types of 3-D shapes, such that 3-D shapes are defined through the placement of a subset of their vertices, instead of explicit definition of all the degrees-of-freedom associated with the objects.

Another advantage of using 3 corners to define a box is that this approach is highly symmetric in the actions required by the user. This implies that the box shaping process can broken down into a series of identical and manageable steps. Indeed, the evaluation of the 3C techniques revealed that the most accurate results are produced when the process is broken down into 3 identical steps, i.e. the 3+3+3 DoF 3C technique.

The evaluation of box shaping techniques was performed as a docking experiment. This implied that some way of assessing the quality of the end result of the docking operation was needed. Since no such quantitative method existed, a new one was introduced. This method was based on finding the minimum combined translation of the corners of one box to the corners of another box. Thus, the method represents a metric on the space of all 3-D boxes. The method proved useful in the evaluation, in that it succeeded in detecting the differences between the different techniques. In terms of generalization, the suggested box distance metric also has potential to be used in other contexts than boxes. For instance, it immediately generalizes to octahedrons, since these shapes share the same symmetry group as boxes. The outlined method for computing the metric can easily

be adapted to shapes with a different symmetry group as well.

The evaluation revealed three main results. The first result is that the 3+3+3 DoF 3C technique is more precise than the tested alternative methods. The second result is that in shaping boxes, techniques that break the definition of DoF into small batches perform better in terms of precision. It is hypothesized that this result is likely to generalize to many other interaction tasks in VE's. The final result is that there seems to be no difference in terms of speed between defining orientation explicitly or implicitly through three corners.

Because of the encouraging results regarding the 3+3+3 DoF 3C box shaping technique, this technique is the preferred choice in the other two papers, which both include box shaping as part of their experimental design.

The second paper, Paper B, makes a small digression from volume shaping since it does not present any new ways of defining a volume. Instead, it deals with techniques for perceiving a discretely sampled volume represented as a 3-D scatterplot. Although it does not introduce new volume shaping techniques, its experiment does require the user to define volumes, specifically boxes. The main idea of Paper B is to render 3-D glyphs such that they always cover the same amount of screen space, or an approximately constant visual angle. Hence the name CVA glyphs. The rationale of CVA glyphs is that, unlike perspective glyphs, they do not obscure an increasing part of the field-of-view, when the user moves close. This has the potential of allowing better perception of structures in cases with a lot of clutter/noise around the structures. Specifically, when CVA glyphs are combined with an otherwise regular perspective view of the virtual scene, they present two useful properties, namely the ability to spread and thin out clutter when the user moves close, and the ability to condense and solidify structures when the user moves away.

There were two goals of the evaluation of CVA glyphs. One was to compare them to regular, perspective glyphs both in cluttered and non-cluttered situations. The other was to see how the volumetric density of the structures in the 3-D scatterplot affected the ability to perceive structures. The main results of that experiment was the finding that CVA glyphs are beneficial in situations with lots of clutter. In situations without clutter, CVA glyphs did not deteriorate the ability to perceive patterns. However, regular perspective glyphs were preferable in cases with very low density. The experiment also uncovered the possible locations of two density thresholds, which significantly increase the ability to perceive structures in a 3-D scatterplot. Specifically, having more than 50,000 glyphs/m^3 made shape perception very easy, while having less than 5,000 glyphs/m^3 greatly impaired shape perception.

The results of Paper B are used in the third paper. Thus, both CVA glyphs and the finding that having more than 50,000 glyphs/m^3 improves pattern perception are applied in the design of the experiments of Paper C.

Paper C concentrates on the task of efficiently selecting a large number of objects. Conceptually, selecting many objects involves the definition of a volume around the desired objects. The paper suggests the introduction of a new technique for this purpose, the magic wand. The philosophy of the magic wand is to get the computer to do most of the work involved in the selection task, such that the user only has to indicate a single sample of interest, which then serves as a

seed for automatically expanding the selection. The magic wand is compared to two existing techniques, a box shaped lasso and a sherical brush.

The evaluation of the three techniques also involved a second factor, the geometric layout of the selection targets, since this was expected to be highly influential wrt. the applicability of the techniques. The results strongly supported this hypothesis. Thus, it is very important to consider the geometric scenario before choosing which multiple object selection tool to use. However, unlike exisiting hypotheses on the issue, the shape of the selection volume does not have to match the shape of the structure of the selection targets. Instead, it seems that using simple tools is preferable to using complex tools that fit the shape of the targets perfectly. This is strong evidence that some basic underlying factor is the real distinguishing factor in assessing the usefulness of a multiple object selection tool.

One interesting observation, which can only be made by collectively inspecting the results of all three papers is the fact that task completion times seem to be less influenced by the choice of volume shaping technique than end result quality. I.e. volume shaping techniques often achieve very different levels of quality in roughly the same amount of time. The exception to this rule is the magic wand technique, which delegates a lot of the work from the user to a computer.

Another interesting point to note is that a logarithm transformation of the completion times in all studies leads to approximately normal distributions. I.e. the completion time data is log-normal distributed. The log-normal distribution is asymmetrical and skewed to the left, meaning that short completion times are much more common that long completion times. Checking other literature about completion times from other fields of study reveals that this is a general tendency, also outside interactions in virtual environments [65, 26]. To some extent, this serves as validation of the chosen experimental approach, in that the results match the tendencies found by numerous other researchers of completely different fields.

Summing up the results, this thesis has contributed to the topic of shaping volumes in several different ways. The thesis has demonstrated new tehniques for shaping 3-D volumes through the 3C box shaping techniques and the magic wand for making a volumetric selection. Furthermore, a glyph rendering technique, CVA, which allows for better perception of volumetric structures in adverse conditions has been introduced. The suggested techniques have all been evaluated through rigourous studies, which have highlighted both pros and cons of the suggested techniques. Although this thesis has to a large extent focused on 3-D boxes, there is a legion of other possible volumetric shapes to create. This, however, does not imply that the lessons learned through the studies of 3-D boxes should not apply to other shapes as well.

One general theme, which has permeated much of the work presented in this thesis is the notion of degrees-of-freedom (DoF). The value and importance of DoF seems to be under-estimated in current VE interaction technique design. One example of the potential of using DoF as design principle is the equivalence of shaping a volume to the definition of that volume's DoF. This idea served as the primary, guiding principle that led to the design of the 3C techniques. Furthermore, the results of the final paper of this thesis strongly indicate that the

DoF associated with multiple object selection tools may be more important than many other design parameters. One viable hypothesis of future work in this area is therefore, that controlling as few DoF as possible to solve a task is preferable to controlling more DoF, even when these additional DoF implies a high degree of expressivity and precision. The notion of DoF is general enough that it is possible that DoF can form the basis of new design methodologies for interaction techniques, or that existing techniques can be evaluated and compared in the light of their DoF.

References

[1] D. Asimov. The grand tour: a tool for viewing multidimensional data. *SIAM J. Sci. Stat. Comput.*, 6:128–143, January 1985.

[2] A. J. Berkhout. A holographic approach to acoustic control. *J. Audio Eng. Soc*, 36(12):977–995, 1988.

[3] A. Bierbaum, C. Just, P. Hartling, K. Meinert, A. Baker, and C. Cruz-Neira. Vr juggler: a virtual platform for virtual reality application development. In *Virtual Reality, 2001. Proceedings. IEEE*, pages 89 –96, march 2001.

[4] W. W. Blessing, A. A. Landauer, and M. Coltheart. The effect of false perspective cues on distance and size-judgments: an examination of the invariance hypothesis. *The American journal of psychology*, 80(2):250–256, 1967.

[5] D. A. Bowman and L. F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Proceedings of the 1997 symposium on Interactive 3D graphics*, I3D '97, pages 35–ff., New York, NY, USA, 1997. ACM.

[6] D. A. Bowman, E. Kruijff, J. J. LaViola, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley, first edition, 2005.

[7] A. Buja, J. A. McDonald, J. Michalak, and W. Stuetzle. Interactive data visualization using focusing and linking. In *Proceedings of the 2nd conference on Visualization '91*, VIS '91, pages 156–163, Los Alamitos, CA, USA, 1991. IEEE Computer Society Press.

[8] M. S. T. Carpendale, T. Carpendale, D. J. Cowperthwaite, and F. D. Fracchia. Distortion viewing techniques for 3-dimensional data. In *Proceedings of the 1996 IEEE Symposium on Information Visualization (INFOVIS '96)*, INFOVIS '96, pages 46–, Washington, DC, USA, 1996. IEEE Computer Society.

[9] A. Cohé, F. Dècle, and M. Hachet. tbox: a 3d transformation widget designed for touch-screens. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 3005–3008, New York, NY, USA, 2011. ACM.

[10] C. Cruz-Neira, D. J. Sandin, T. A. DeFanti, R. V. Kenyon, and J. C. Hart. The cave: audio visual experience automatic virtual environment. *Commun. ACM*, 35:64–72, June 1992.

[11] G. De Haan, M. Koutek, and F. H. Post. IntenSelect: Using Dynamic Object Rating for Assisting 3D Object Selection. In *In Virtual Environments 2005*, volume 2005, pages 201–209, 2005.

[12] A. W. Donoho, D. L. Donoho, and M. Gasko. Macspin: Dynamic graphics on a desktop computer. *IEEE Comput. Graph. Appl.*, 8:51–58, July 1988.

[13] N. Elmqvist, P. Dragicevic, and J. D. Fekete. Rolling the Dice: Multidimensional Visual Exploration using Scatterplot Matrix Navigation. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1539–1148, Nov. 2008.

[14] C. Ericson. *Real-Time Collision Detection*. Morgan Kaufmann Publishers, first edition, 2005.

[15] B. Fröhlich. The quest for intuitive 3-d input devices. 2005.

[16] R. Fuchs, V. Welker, and J. Hornegger. Non-convex polyhedral volume of interest selection. *Computerized Medical Imaging and Graphics*, 34(2):105 – 113, 2010.

[17] M. A. Gerzon. Periphony: With-height sound reproduction. *J. Audio Eng. Soc*, 21(1):2–10, 1973.

[18] E. Gokcay and J. C. Principe. Information theoretic clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:158–171, 2002.

[19] E. Granum and P. Musaeus. *Constructing virtual environments for visual explorers*, pages 112–138. Springer-Verlag, London, UK, 2002.

[20] D. Hammershøi and H. Møller. *Binaural Technique*, pages 223–254. 2005.

[21] K. Hinckley, J. Tullio, R. Pausch, D. Proffitt, and N. Kassell. Usability analysis of 3d rotation techniques. In *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*, pages 1–10, New York, NY, USA, 1997. ACM.

[22] A. Huckauf, A. Speed, A. Kunert, J. Hochstrate, and B. Fröhlich. Evaluation of 12-dof input devices for navigation and manipulation in virtual environments. In M. F. Costabile and F. Paternò, editors, *Human-Computer Interaction - INTERACT 2005*, pages 601–614. Springer Berlin / Heidelberg, 2005.

[23] A. Huckauf, A. Speed, A. Kunert, J. Hochstrate, and B. Fröhlich. Evaluation of 12-dof input devices for navigation and manipulation in virtual environments. In M. F. Costabile and F. Paternò, editors, *Human-Computer Interaction - INTERACT 2005*, pages 601–614. Springer Berlin / Heidelberg, 2005.

[24] J. Jang and J. R. Rossignac. Multiple object selection in pattern hierarchies. Technical report, Georgia Institute of Technology, 2007.

[25] J.-S. Kim, D. Gracanin, K. Matkovic, and F. Quek. Finger walking in place (fwip): a traveling technique in virtual environments. In *Proceedings of SmartGraphics 2008, Springer LNCS 5166/2008*, LNCS. Springer, August 2008.

[26] S. Kochhar, S. Mazzocchi, and P. Paritosh. The anatomy of a large-scale human computation engine. In *HCOMP '10: Proceedings of the ACM SIGKDD Workshop on Human Computation*, pages 10–17, New York, NY, USA, 2010. ACM.

[27] K. Koffka. *Principles of Gestalt Psychology*, volume 7. Harcourt, Brace and World, 1935.

[28] R. Kopper, F. Bacim, and D. A. Bowman. Rapid and accurate 3d selection by progressive refinement. *3D User Interfaces*, 0:67–74, 2011.

[29] J. B. Kuipers. *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace and Virtual Reality*. Princeton University Press, Aug. 2002.

[30] M. Latoschik and H. Tramberend. Simulator x: A scalable and concurrent architecture for intelligent realtime interactive systems. In *Virtual Reality Conference (VR), 2011 IEEE*, pages 171 –174, march 2011.

[31] Y. Li. Oriented particles for scientific visualization. Master's thesis, University of New Brunswick, Faculty of Computer Science, 1997.

[32] J. F. Lucas. Design and evaluation of 3d multiple object selection techniques. Master's thesis, Virginia Polytechnic Institute and State University, 2005.

[33] J. F. Lucas, J.-S. Kim, and D. A. Bowman. Resizing beyond widgets: object resizing techniques for immersive virtual environments. In *CHI '05 extended abstracts on Human factors in computing systems*, CHI EA '05, pages 1601–1604, New York, NY, USA, 2005. ACM.

[34] D. E. Manolakis. Efficient solution and performance analysis of 3-D position estimation by trilateration. *Aerospace and Electronic Systems, IEEE Transactions on*, 32(4):1239–1248, 1996.

[35] M. R. Masliah and P. Milgram. Measuring the allocation of control in a 6 degree-of-freedom docking experiment. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 25–32, New York, NY, USA, 2000. ACM.

[36] A. Mazeika, M. H. Böhlen, and P. Mylov. Using nested surfaces for visual detection of structures in databases. In *Visual Data Mining*, pages 91–102. 2008.

[37] M. Mine. Virtual environment interaction techniques. Technical report, UNC Chapel Hill CS Dept, 1995.

[38] M. R. Mine, F. P. Brooks, Jr., and C. H. Sequin. Moving objects in space: exploiting proprioception in virtual-environment interaction. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 19–26, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[39] D. C. Montgomery. *Design and Analysis of Experiments*. J. Wiley, seventh edition, 2008.

[40] H. R. Nagel. *Exploratory Visual Data Mining in Spatio-Temporal Virtual Reality*. PhD thesis, Faculty of Engineering and Science, Aalborg University, 2005.

[41] H. R. Nagel, E. Granum, S. Bovbjerg, and M. Vittrup. Immersive visual data mining: The 3dvdm approach. In *Visual Data Mining*, pages 281–311. 2008.

[42] M. Nakao, T. Watanabe, T. Kuroda, and H. Yoshihara. Interactive 3d region extraction of volume data using deformable boundary object. *Stud Health Technol Inform*, 111:349–52, 2005.

[43] G. Norman. Likert scales, levels of measurement and the "laws" of statistics. *Advances in health sciences education theory and practice*, 15(5):625–632, 2010.

[44] K. R. Popper. *The Logic of Scientific Discovery*. Hutchinson, London, 1934.

[45] K. R. Popper. *Conjectures and refutations: The growth of scientific knowledge*. Harper & Row, New York, 1968.

[46] I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa. The go-go interaction technique: non-linear mapping for direct manipulation in vr. In *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 79–80. ACM, 1996.

[47] D. Reiners. *OpenSG: a scene graph system for flexible and efficient realtime rendering for virtual and augmented reality applications*. PhD thesis, Technischen Universität Darmstadt, 2002.

[48] H. Sanftmann and D. Weiskopf. Illuminated 3d scatterplots. *Comput. Graph. Forum*, 28(3):751–758, 2009.

[49] R. Sibson. SLINK: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, Jan. 1973.

[50] A. Steed. Towards a general model for selection in virtual environments. In *3D User Interfaces, 2006. 3DUI 2006. IEEE Symposium on*, pages 103 – 110, march 2006.

[51] A. Steed and C. Parker. 3D Selection Strategies for Head Tracked and Non-Head Tracked Operation of Spatially Immersive Displays. In *8th International Immersive Projection Technology Workshop*, Ames, Iowa, USA, May 2004.

[52] S. S. Stevens. On the Theory of Scales of Measurement. *Science*, 103(2684):677–680, 1946.

[53] W. Stuerzlinger and G. Smith. Efficient manipulation of object groups in virtual environments. In *Virtual Reality, 2002. Proceedings. IEEE*, pages 251 –258, 2002.

[54] E. Suma, S. Clark, S. Finklestein, Z. Wartell, D. M. Krum, and M. Bolas. Leveraging change blindness for redirection in virtual environments. In *IEEE Virtual Reality*, pages 159–166, Mar. 2011.

[55] I. E. Sutherland. A head-mounted three dimensional display. In *Fall Joint Computer Conference, 1968*, pages 757–764, 1968.

[56] A. Treisman. Perceptual grouping and attention in visual search for features and objects. *Journal of Experimental Psychology: Human Perception and Performance*, 8:194–214, 1982.

[57] E. R. Tufte. *The Visual Display of Quantitative Information, 2nd edition*. Graphics Press, 2 edition, May 2001.

[58] A. Ulinski, Z. Wartell, P. Goolkasian, E. A. Suma, and L. F. Hodges. Selection performance based on classes of bimanual actions. In *IEEE Symposium on 3D User Interfaces*, pages 51–58, 2009.

[59] A. Ulinski, Z. Wartell, and L. F. Hodges. Bimanual task division preferences for volume selection. In *VRST '07: Proceedings of the 2007 ACM symposium on Virtual reality software and technology*, pages 217–218, New York, NY, USA, 2007. ACM.

[60] A. Ulinski, C. Zanbaka, Z. Wartell, P. Goolkasian, and L. Hodges. Two handed selection techniques for volumetric data. *3D User Interfaces, 2007. 3DUI '07. IEEE Symposium on*, March 2007.

[61] A. C. Ulinski. *Taxonomy and experimental evaluation of two-handed selection techniques for volumetric data*. PhD thesis, University of North Carolina at Charlotte, Charlotte, NC, USA, 2008. Adviser-Hodges, Larry F.

[62] A. van Dam. Post-wimp user interfaces. *Commun. ACM*, 40:63–67, February 1997.

[63] J. Viega, M. J. Conway, G. Williams, and R. Pausch. 3d magic lenses. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*, UIST '96, pages 51–58, New York, NY, USA, 1996. ACM.

[64] Y. Visell. *Walking in virtual worlds: Physics, Perception, and Interface Design*. PhD thesis, McGill University, Dept. of Electrical and Computer Engineering, 2011.

[65] J. Wang, O. Alonso, and P. G. Ipeirotis. Estimating the completion time of crowdsourced tasks using survival analysis models. *Search*, (Csdm):1–38, 2011.

[66] N. Wang, A. Paljic, and P. Fuchs. *A study of Perception of volumetric rendering for immersive scientific visualization*, pages 145—152. 2010.

[67] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, second edition, 2004.

[68] C. Ware, K. Arthur, and K. S. Booth. Fish tank virtual reality. In *Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems*, CHI '93, pages 37–42, New York, NY, USA, 1993. ACM.

[69] G. Welch and E. Foxlin. Motion tracking: No silver bullet, but a respectable arsenal. *IEEE Comput. Graph. Appl.*, 22:24–38, November 2002.

[70] G. Wills. Selection: 524,288 ways to say "this is interesting". In *Information Visualization '96, Proceedings IEEE Symposium on*, pages 54 –60, 120, oct 1996.

[71] R. C. Zeleznik, K. P. Herndon, and J. F. Hughes. Sketch: an interface for sketching 3d scenes. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[72] S. Zhai. *Human Performance in Six Degree of Freedom Input Control*. PhD thesis, University of Toronto, Toronto, Canada, 1995.

[73] S. Zhai, W. Buxton, and P. Milgram. The "silk cursor": investigating transparency for 3d target acquisition. In *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence*, CHI '94, pages 459–464, New York, NY, USA, 1994. ACM.

[74] S. Zhai and P. Milgram. Quantifying coordination in multiple dof movement and its application to evaluating 6 dof input devices. In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 320–327. ACM Press/Addison-Wesley Publishing Co., 1998.

[75] D. Zotkin, J. Hwang, R. Duraiswaini, and L. Davis. Hrtf personalization using anthropometric measurements. In *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, pages 157 – 160, oct. 2003.

[76] E. V. Zudilova and P. M. A. Sloot. Virtual reality and desktop as a combined interaction-visualisation medium for a problem-solving environment. In *Proceedings of the 2003 international conference on Computational science: PartIII*, ICCS'03, pages 1025–1034, Berlin, Heidelberg, 2003. Springer-Verlag.

Contributions

Paper A: Shaping 3-D Boxes: A Full 9 Degree-of-Freedom Docking Experiment

Paper B: Shape Perception in 3-D Scatterplots Using Constant Visual Angle Glyphs

Paper C: Brush, Lasso, or Magic Wand? Picking the Right Tool for Large-Scale Multiple Object Selection Tasks

Paper A

Shaping 3-D Boxes: A Full 9 Degree-of-Freedom Docking Experiment

Rasmus Stenholt and Claus B. Madsen

This paper was presented and published as a long paper in:
Proceedings of the IEEE VR 2011.

Abstract

Enabling users to shape 3-D boxes in immersive virtual environments is a non-trivial problem. In this paper, a new family of techniques for creating rectangular boxes of arbitrary position, orientation, and size is presented and evaluated. These new techniques are based solely on position data, making them different from typical, existing box shaping techniques. The basis of the proposed techniques is a new algorithm for constructing a full box from just three of its corners. The evaluation of the new techniques compares their precision and completion times in a 9 degree-of-freedom (DoF) docking experiment against an existing technique, which requires the user to perform the rotation and scaling of the box explicitly. The precision of the users' box construction is evaluated by a novel error metric measuring the difference between two boxes. The results of the experiment strongly indicate that for precision docking of 9 DoF boxes, some of the proposed techniques are significantly better than ones with explicit rotation and scaling. Another interesting result is that the number of DoF simultaneously controlled by the user significantly influences the precision of the docking.

1 Introduction

Rectangular, 3-dimensional boxes, or more simply put, boxes, are very commonly found in both real and virtual environments. Prominent examples of boxes from real environments are bricks, storage crates, and dice. In desktop 3-D modeling applications such as 3ds Max [1], the box is often one of the primary building blocks of more complex models. Indeed, in 3ds Max, the first option among the standard primitives is the box. Boxes are also frequently used in virtual environments as bounding volumes [5] for more complex geometry, e.g. in applications that require real-time collision detection. Deformable bounding boxes may also be useful in designing interaction techniques that require non-rigid manipulation of virtual objects. Furthermore, virtual boxes can also be used for selection tasks, as e.g. presented in [17, 10]. Thus, precise and fast box shaping may be useful as part of several important 3DUI tasks. In the example of selection, precise boxes are important to avoid false selections in situations where the objects to be selected are spatially closely located to undesired objects. For these reasons, it seems important to understand how boxes can be precisely and quickly created by a user in an immersive virtual environment.

To demonstrate the potential usefulness and the complexity involved in shaping an arbitrary 3-D box, it is practical to compare it to a similar, but less complex operation: The creation of axis-aligned 2-D boxes, or rectangles. Such rectangles are frequently shaped by users of desktop computers. The most common example of this is probably the rectangle created to select multiple items in a desktop

environment. This operation can be carried out using a single hand, by clicking at a desired corner of the selection rectangle, followed by dragging away the opposite corner of the rectangle to adjust the rectangle's dimensions. The term axis-aligned refers to the fact that all created rectangles share a common orientation, typically aligned to the edges of the screen. Axis-aligned rectangle construction is possible using only one hand, since such a rectangle only has 4 DoF, in this case given by the 2-D positions of two opposite corners. These corners can be input sequentially by a single hand controlling a mouse. The input is sequential since it has three distinct steps: 1) Indicate the first corner by clicking a mouse button, 2) drag the mouse to the desired second corner, 3) release the mouse button to place the second corner. Removing the axis-alignment constraint would add an additional DoF to allow for rotation of the rectangle, yielding a total of 5 DoF.

If these ideas are extended to creation of 3-D boxes, the case becomes somewhat more complex. First of all, the box itself now requires independent specification of a 3-D position, a 3-D orientation and 3 size parameters, yielding a total of 9 DoF for an arbitrary box. In the axis-aligned case, this number of DoF reduces to 6 (i.e. a 3-D position and 3 size parameters). Secondly, if the operation is performed in a head-tracked virtual environment, the user's own motions also need to be controlled while creating the box.

As this paper will demonstrate, the 9 box parameters do not have to be given as an explicit combination of position, orientation, and size. Rather, a new family of techniques where the shape of a box is specified based on 3 positions only is introduced and tested. The user task chosen for the test is docking, since this decouples the box shaping task from the specific use of the box while still providing some ground truth data to measure precision against. Thus, the aim of this paper is to investigate the precision and completion time for users producing arbitrary 3-D boxes in a head-tracked virtual environment, exploring various ways of allocating the user's control of the 9 parameters required for an arbitrary box, i.e. the mapping of user control to box parameters.

2 Related Works

2.1 3-D Boxes in Object Selection

Few studies have been made about shaping 3-D boxes in immersive virtual environments. In all cases, these studies have focused on shaping boxes for selection of single or multiple virtual objects. Even in very comprehensive surveys of 3-D user interaction, such as [3], selection boxes and interaction with multiple objects is left untreated.

In a series of studies [18, 17, 16], Ulinski et al, have tested three different techniques, named Hand-on-Corner (HOC), Hand-in-Middle (HIM) and Two Corners (TC), for box creation. The context of these studies is two-handed selection of volumetric data. The tests took place in a desktop environment, where the user was sitting in front of a screen, manipulating the boxes using magnetic 6 DoF tracking equipment. In HOC, the user's non-dominant hand holds one corner of the box, effectively controlling its position and orientation, i.e. 6 DoF is measured at the non-dominant hand. The dominant hand controls the position

of the opposite corner. In DoF terms this corresponds to controlling the three size parameters of the box. The HOC and HIM techniques differ only by the position of the user's non-dominant hand relative to the box. Both of these techniques allow for full 9 DoF manipulation of the box with orientation given explicitly by the user. For this reason, HOC-like techniques will serve as the counterpart to the techniques proposed in this study.

In TC, the user shapes a box by dragging apart two diagonally opposite corners of the desired box. Although only controlling two corners of a box may seem more attractive than the proposed three corners of this paper, such a scheme is problematic with respect to the DoF of the constructed box: Since the shape of a TC box depends only on the 6 independent parameters given by the 3-D positions of two hands (both position, orientation, and scale are computed simultaneously from these 6 parameters), the created boxes cannot feature 9 independent DoF. Rather, some interdependency between orientation and scale must result from this approach. Alternatively, the TC technique may be used as an axis-aligned box construction algorithm. It is not clear from the sources describing TC, however, if anything has been done to overcome this problem. The mentioned studies have used bimanual symmetry/asymmetry and synchronicity/asynchronicity of actions as the main experimental factors. The results of these studies indicate that synchronous, symmetric interaction (in that case TC) outperforms other techniques with respect to both accuracy and completion time, indicating that the techniques proposed in this paper must perform well against HOC techniques.

The studies of Ulinski et al differs from the approach taken in this paper in several respects. In this paper, the focus is not on boxes specifically for selection, but on box construction as a generic task in its own right. This fact changes the user task of the experiment from selection to docking. Furthermore, the user is placed inside the virtual test environment using a head-mounted display (HMD). This allows the user to take advantage of the 3-D information gained from stereopsis and self-motion. The factors of this study focus on the allocation and types of DoF to the hands, not the synchronicity or symmetry of the hands' control, although the tested techniques must implicitly incorporate different levels of these factors as well. For illustration, a small example of the scene exposed to the test subjects is presented in Figure A.1.

In [10], a box shaping technique is introduced for multiple object selection. The proposed technique in that study uses a persistent box that the user can manipulate in two ways. Position and orientation are controlled using the Go-Go technique [13], whereas scaling is performed using a technique named PORT in which the user aligns a cursor with the dimension of the box to be scaled. This implies that the action of scaling has been separated from the other manipulations, and that scaling occurs along one axis at a time. The Go-Go+PORT technique was not compared against any other box shaping methods.

In [21], Zhai studied the effects of using semi-transparency for a fixed-shape selection box. The results indicated that semi-transparent boxes outperformed wireframe renderings for selection tasks, where the goal was to get a single object to be completely contained inside the selection box. It is not known if this result would also apply to docking tasks. However, this will not be studied further in this paper.

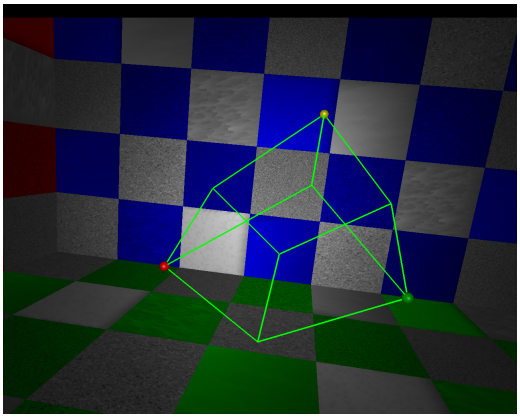

Figure A.1: A box being shaped using one of the techniques proposed in this paper. The corner marked by a yellow sphere has already been locked. The user is currently controlling the red (non-dominant hand) and green (dominant hand) spheres to complete the box.

The standard method of creating a box in 3-D modelling applications also deserves to be mentioned in this context. The typical operation of shaping a 3-D box using a mouse, as e.g. seen in 3ds Max [1], is a process consisting of several steps. First, the base rectangle of an axis-aligned box is traced out, giving two of the final box' dimensions. Then the 3rd dimension of the box is extruded orthogonally to the base. Finally, rotation and translation transformations can be applied separately to place the box at the desired location/orientation. Although this specific technique will not be examined in this study, its separation of DoF specification into several steps to overcome the limitations of the input device has served as inspiration.

2.2 Docking

The task of docking, i.e. matching the spatial characteristics of one virtual object to another, has been frequently studied. However, docking a deformable object which increases the number of DoF above the 6 required for the rigid transformations of translation and rotation, does not seem to have been the subject of any other study yet. This makes the choice of a 9 DoF docking task presented in this paper interesting, since it may provide some new insights.

In [21, 22] several experiments relating to 6 DoF docking and tracking are presented. One of the results indicate that users tend to focus on subsets of DoF when performing 6 DoF tracking tasks. This is also one of the findings of [11], which also notes that users typically switch between translational and rotational modes of manipulation. In an experiment about matching the orientation of a house to an identical target using several different techniques [7], the authors note that some users have difficulties specifying orientation. This supports the main idea of the techniques proposed in this paper, where orientation is not ex-

plicitly specified by the user.

There are many examples of interaction devices that feature more than 6 DoF, e.g. the CubicMouse, the YoYo, and the SquareBone. In [8], such devices are tested in a docking experiment, where the users utilized the additional DoF to navigate a VE with the goal of docking a subset of a cube, implying that the actual docking was only a 6 DoF task.

2.3 Bimanual Interaction

Although the techniques introduced in this study are not dependent on using both hands, the fact that the user has to produce input data for 9 DoF makes it possible, and probably useful, to let the user put both hands to use in the docking task. The body of literature on bimanual interaction is quite large. Some studies have focused on understanding and modeling the actions and interactions of the hands [6], whereas others have focused on specific applications of bimanual input such as selection [4, 18, 2] or modeling [9, 19]. A common conclusion among many of these studies is that bimanual input is superior to single-handed input in many contexts. It is also often stated, as e.g. summarized in [20], that the operations of the non-dominant hand should ideally provide a frame of reference for the dominant hand's more precise manipulations. This result is used in the design of HOC techniques, where orientation is mapped to the non-dominant hand.

3 Theory of Boxes from 3 Corners

Data representations of arbitrary 3-D boxes can be classified by the number of parameters that they contain. If a representation contains exactly 9 independent parameters, it belongs to the class of minimal box representations. The HOC techniques are based on such a representation. If a representation has less than 9 independent parameters, the representation can be classified as sub-minimal, and thus ambiguous, in that infinitely many boxes will match the given parameters. This is e.g. the case for the TC technique of [17]. If more than 9 parameters are stated, then the representation is excessive. Such representations are e.g. presented in [5] for use as collision geometry. For scenarios where users are going to construct boxes, the minimal and possibly sub-minimal classes are potentially the most interesting, since they require the user to produce the least amount of data.

The parameters defining a box can be given explicitly as the translation, rotation, and scaling of the represented box. This is the foundation of the HOC technique described in [17]. As suggested in this paper, an alternative way of getting the 9 parameters is from the combination of three 3-D positions corresponding to three corners of the desired box, thus eliminating the need for direct orientation input. Such techniques will collectively be referred to as 3-corner (3C) techniques. Before describing the performed experiment, this section will elaborate on the technical details of 3C boxes.

There are three distinct ways of choosing 3 corners on a box:

1. 3 corners, A, B, C, on the same face of the box, producing a right triangle from two box edges and a face diagonal.

2. 2 neighbouring corners, A, B, and another, C, across a box diagonal from A or B, producing a right triangle from a box edge, a face diagonal, and a box diagonal.

3. 3 corners, A, B, C, sharing a common neighbouring corner, producing an acute triangle, tracing out 3 connected face diagonals.

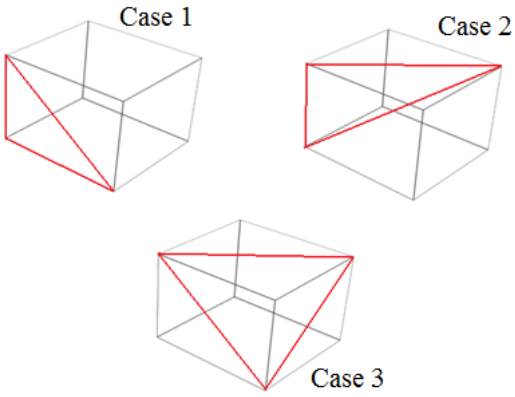

Figure A.2: Illustration of the 3 cases of picking 3 corners on a box. Case 3 is the one of interest for the techniques presented in this paper.

The cases are also illustrated in Figure A.2. Cases 1 and 2 above are problematic because the triangles formed are always right triangles, meaning that one of their angles is fixed. This fact lowers the number of DoF in the representation to 8. In terms of boxes, this means that infinitely many boxes can be constructed from such combinations of 3 corners. Note that the TC technique of [18] is a partial case 2 representation, with the right angle corner still unspecified.

This leaves only case 3, in which the triangle formed by the given points is always acute. Acute triangles need 3 parameters to be completely specified in a given position and orientation, but the range of these parameters is limited, i.e. all angles must be in the open interval $]0°; 90°[$. In other words, the 9 parameters are independent inside an open subset of the space of all triangles. This provides enough independence to ensure that only 2 boxes are represented by any acute triangle, which makes it possible to consistently pick one of the two, making it a minimal representation.

The difficult parts of a 3C representation are thus: 1) Find the common neighbour, $D = [d_x, d_y, d_z]^T$, of corners $A = [a_x, a_y, a_z]^T$, $B = [b_x, b_y, b_z]^T$, $C = [c_x, c_y, c_z]^T$ to allow for simple reconstruction of the remaining box. 2) Find a disambiguation rule that picks one of the two possible solutions for D.

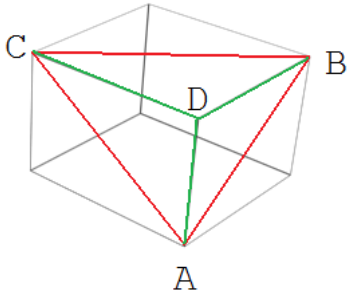

Figure A.3: To construct a box from a 3C representation, the coordinates of the point D must be determined. The 3-D shape formed by $ABCD$ is a trirectangular tetrahedron.

3.1 Finding the 4th Corner

A geometric view of the 4th corner, D, is illustrated in Figure A.3. Analytically, D is found by solving the following non-linear system of equations[1], which is based on the desired orthogonality of the box edges:

$$\left\{ \begin{array}{l} (D-A)^T(D-B)=0 \\ (D-B)^T(D-C)=0 \\ (D-A)^T(D-C)=0 \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{l} D^TD-(A+B)^TD+A^TB=0 \\ D^TD-(B+C)^TD+B^TC=0 \\ D^TD-(A+C)^TD+A^TC=0 \end{array} \right\} \tag{A.1}$$

The D^TD terms in the system implies that the equations are three quadratic polynomials in three unknowns, d_x, d_y, d_z, the coordinates of the desired point, D. Since the coefficents of the quadratic terms are the same in all three equations, the system reduces to a single quadratic and two linear equations. The consequences of this are twofold: 1) There will only be 2 solutions for D. 2) D cannot be determined purely by linear algebra methods such as Gauss-elimination. The direct solution expressed in terms of the coordinates of A, B, C to the system is very long. This expression e.g. requires accessing the coordinates of each point 41946 times. Instead it is proposed to solve the system by the following procedure:

1. Reduce the system to a single quadratic equation and two linear equations by subtracting the first equation from the second and third.

2. Reduce the two linear equations using Gauss-elimination to an augmented matrix of the following form.

$$\left[\begin{array}{cccc} 1 & 0 & k & l \\ 0 & 1 & m & n \end{array} \right] \tag{A.2}$$

[1]It is possible to rearrange the system of equations to reveal that it is, in fact, equivalent to intersecting three spheres. These spheres are defined by each having an edge of triangle ABC as diameter. This means that the system can also be solved by the 3-D trilateration methods used in GPS systems and surveying. This approach has not been tested.

3. Substitute d_x and d_y in terms of the the intermediate variables k, l, m, n into the quadratic equation and solve for d_z.

4. Substitute the two solutions for d_z into the linear equations and solve for d_x and d_y.

The steps above only need to be performed once symbolically to arrive at a smaller, analytical solution expression. Although the runtime complexity of both suggested solutions are constant time, $O(1)$, since the size of the input is always the same, the time constants of the two are very different, making the second method a worthwhile optimization. In a small-scale performance test where a specific system was solved 100,000 times in a row using both methods, the solution in terms of intermediate variables was, on average, app. 420 times faster to compute than the direct solution.

3.2 Picking a Solution

Having determined the two candidates for the 4th point, D, on the box, the next step is to create a heuristic to disambiguate between the two. Examining the geometry of the solutions, D_1, D_2, reveals that they are mirror images of each other through the plane of triangle ABC. This means that walking the edges of triangle ABC seen from D_1 or D_2 will be a clockwise (CW) walk for one solution and a counterclockwise (CCW) walk for the other one. This provides one possible disambiguation mechanism. The direction of the walk can be expressed mathematically as the sign of the determinant predicate, Δ_{ABCD_i} [5]:

$$\Delta_{ABCD_i} = \begin{vmatrix} a_x & a_y & a_z & 1 \\ b_x & b_y & c_z & 1 \\ c_x & c_y & b_z & 1 \\ d_{ix} & d_{iy} & d_{iz} & 1 \end{vmatrix} \tag{A.3}$$

Another way of disambiguating would be to always choose a particular solution to the quadratic in the reduced equation system. Specifically, this would mean choosing between the two solutions for d_z based on the sign in front of the square root term in the quadratic solution.

In a scenario with a head-tracked user, another solution is available: Use the information given by the user's current head position to disambiguate, e.g. by choosing the D that is closest to the user's head. This is the approach that has been used in the current evaluation of 3C techniques. The main advantages of this approach are that it decouples the disambiguation completely from the sequence of definition of A, B, C and from the layout of the virtual environment's coordinate system. The disadvantage is that the user's motions relative to the box can cause it to suddenly flip between the two solutions. Although the heuristic used has not posed a great problem to the participants in this study, the usability aspects of the choice of disambiguation heuristic may be an interesting factor to include in future studies of 3C techniques.

3.3 Comparing Two Arbitrary Boxes

Since one of the aims of the study is to test the precision of the docking operations performed by the users, it becomes necessary to introduce some measure of the error between a user created box and a target box that the user box is supposed to dock into. Existing methods for comparing two boxes all deal with the question of overlap or collision. [5] has a comprehensive survey of such methods. As such they do not try to quantify the similarity of the boxes.

The proposed solution to this problem is to define a metric on the space of 9 DoF boxes, by computing the minimum translation of each corner on one box to the best-matching, corresponding corner on the other box, and adding up these distances to produce the metric. Finding the minimum translation of corners is not as simple as matching each corner on a box to its closest neighbour on the other box. That approach only works as intended if the two boxes are close to identical to begin with.

Instead, we propose to use two 4×4 transformation matrices, $\boldsymbol{T}_{\beta_1}$ and $\boldsymbol{T}_{\beta_2}$, that transform a standard cube centered at the origin into the two boxes, β_1 and β_2, respectively. Then, the minimum distance, $\delta_(\beta_1, \beta_2)$, can be found by the following procedure:

1. Put all 8 corners of β_1 expressed as homogeneous coordinate column vectors into the 4×8 matrix, $\boldsymbol{X}_{\beta_1}$.

2. For each of the 48 possible 4×4 signed permutation matrices [2], $\boldsymbol{P}_i$, in the symmetry group for boxes [15], compute the i^{th} error matrix $\boldsymbol{U}_i$:

$$\boldsymbol{U}_i = \boldsymbol{T}_{\beta_2} \boldsymbol{P}_i \boldsymbol{T}_{\beta_1}^{-1} \boldsymbol{X}_{\beta_1} - \boldsymbol{X}_{\beta_1} \tag{A.4}$$

3. Compute and add up the L^2 norms of the column vectors of $\boldsymbol{U}_i$ to get the i^{th} candidate distance, $\delta_i(\beta_1, \beta_2)$ between the boxes. Stated in linear algebra terms:

$$\delta_i(\beta_1, \beta_2) = \sum_{j=1}^{8} \sqrt{\left(\boldsymbol{U}_i^T \boldsymbol{U}_i\right)_{j,j}} \tag{A.5}$$

4. Choose the minimum $\delta_i(\beta_1, \beta_2)$ to be the distance between the boxes, $\delta_(\beta_1, \beta_2)$.

The logic of the error matrix computation is the following: First, one of the boxes, in this case β_1 is transformed to a standard, origin-centered cube using $\boldsymbol{T}_{\beta_1}^{-1}$. This transformation only has to be done once, since it is the same for all 48 permutations. Second, the corners of the standard cube are permuted using $\boldsymbol{P}_i$ to account for the i^{th} possible symmetry relation of a box. Third, the permuted standard cube is transformed into β_2 using $\boldsymbol{T}_{\beta_2}$. Finally, the original β_1 is subtracted from its current mapping to β_2. This produces the error matrix, $\boldsymbol{U}_i$, that contains

[2] A signed permutation matrix is a matrix with just one non-zero entry in each column and row, where the non-zero entries can only be 1 or -1. In this specific context, it is always the case that $\boldsymbol{P}_{4,4} = 1$.

the i^{th} candidate set of translation vectors from the corners of β_1 to some possible counterparts of β_2. The final metric is then just the candidate translation set that produces the minimum combined translation. It is worth noting that $\delta(\beta_1, \beta_2)$ is a metric function, since it is just a minimized sum of 8 Euclidean distances in $\mathbb{R}^3$. The matrix $\boldsymbol{T}_{\beta_2}\boldsymbol{P}_i\boldsymbol{T}_{\beta_1}^{-1}$, which produces the minimum error, can be decomposed into separate scale, rotation, and translation matrices to diagnose the causes of any error. This feature will not been used in this paper, however.

4 Experimental Procedure

4.1 Experimental Factors and Hypotheses

The 3C techniques differ only by the number of DoF, i.e. corners, to be defined by the user simultaneously. Since a 3C box requires three corners to be fully defined, and each corner locks 3 DoF, the allocation of DoF to each hand gives rise to the following four techniques in the 3C family:

3+3+3 A single corner (3 DoF) is locked each time the user indicates so, e.g. by clicking a button. This implies that the user only needs to work with one hand at any given time. Three clicks will be required by the user to complete a box.

3+6 First a single corner (3 DoF) is placed, after which the two remaining corners (6 DoF) must be placed simultaneously. This technique requires the user to first use one hand, and then synchronously use both hands to complete a box.

6+3 In this technique the user indicates the two end points (6 DoF) of a face diagonal on the desired box simultaneously. The user then indicates the final corner (3 DoF) to complete the box. In this technique the user must use both hands at first, then just one hand to complete a box.

9 Defining 9 simultaneous DoF in a 3C technique means to define 3 corner positions simultaneously. This implies that the two hands of the user are insufficient. In the current experiment, a point offset from the position of the user's head serves as the 3rd corner.

The split between 3+3+3, 3+6, 6+3, and 9 DoF makes up one of the experimental factors with four levels. The other factor is whether the technique belongs to the 3C or HOC family, i.e. two levels. To make a complete test design, the HOC family must have the same four DoF levels defined. This produces the following family of four HOC techniques:

3+3+3 The DoF are locked in batches of three. First, the position (3DoF) of an arbitrary corner is indicated by the user. To make the technique similar to the original HOC techniques, this is done by the non-dominant hand. After the first corner is placed, the non-dominant hand proceeds to rotate the box (3 DoF) around the fixed box corner. Once the orientation is fixed, the user's dominant hand places the final corner, giving the size of the box (3 DoF).

3+6 First a single corner (3 DoF) is placed by the non-dominant hand. Then the user adjusts the orientation and the scale of the box simultaneously (6 DoF). The orientation of the non-dominant hand controls the box' orientation, while the position of the other hand controls scale.

6+3 With this technique, the user finishes the work of the non-dominant hand simultaneously, i.e. position and orientation (6 DoF), before proceeding with scaling the box (3 DoF) using the position of the dominant hand. This technique is very similar to the one presented as asymmetric/asynchronous in [18].

9 This is the original HOC presented as asymmetric/synchronous in [18]. In this technique, the user must position and orient the box using the non-dominant hand at the same time as the dominant hand scales the box (9 DoF).

There is a, perhaps subtle, difference between the techniques regarding the possibility of seeing the current resulting box. For the 3C 3+3+3, 3+6, and 6+3 techniques, no visual feedback about the possible final appearance of the box can be given until the user makes the first indication to lock some DoF. For 9 DoF 3C and all the HOC techniques, the user can see the resulting box at all times.

An interesting point to note with respect to symmetry and synchronicity, is that it is impossible to design a completely synchronous and symmetric (SS) bimanual technique for creating a box, since this would require an even split of the 9 DoF, i.e. $4\frac{1}{2}$ DoF per hand, which does not make sense. A couple of the 3C techniques, 3+6 and 6+3, feature a single-handed step along with a SS bimanual step. The 9 DoF 3C technique is completely SS, since it incorporates the use of a third limb, in this case the user's head. In spite of being SS, the 9 DoF 3C technique is not expected to perform well, since it seems quite unnatural and straining to precisely position the head and hands simultaneously. The technique has been included in the study to be able to make a balanced experimental design with respect to both experimental factors. The two experimental factors and the levels of each are summarized in Table A.1.

Table A.1: The two experimental factors, technique family and DoF allocation, and their associated levels. The entries in the table are read as p=position, o=orientation, s=scale. & means simultaneous defintion, $\rightarrow$ means sequential definition.

Family	DoF allocation			
	3+3+3	3+6	6+3	9
3C	p$\rightarrow$p$\rightarrow$p	p$\rightarrow$p&p	p&p$\rightarrow$p	p&p&p
HOC	p$\rightarrow$o$\rightarrow$s	p$\rightarrow$o&s	p&o$\rightarrow$s	p&o&s

Based on literature and the characteristics of the designed techniques, the three main hypotheses of the experiment are:

H1 3C techniques (with the possible exception of 9 DoF 3C) outperforms HOC techniques in terms of precision.

H2 3C techniques are faster to use than HOC.

H3 Allowing the user to focus on fewer DoF simultaneously increases the precision of the docking.

Hypotheses H1 and H2 are mainly grounded in the results of [18], which states that symmetric techniques in most cases are more precise than asymmetric techniques for box construction. H3 is in line with the findings of [21, 11], where users tended to focus on few DoF simultaneously.

4.2 Experimental Equipment and Method

The experiment was made using an nVisor SX111 HMD[3], which features a total horizontal field-of-view of 102° at a resolution of 1280×1024 pixels per eye. Head and hand tracking was performed using an optical tracking system[4] with 24 infrared cameras spread out around a floor space of 4.5×4.5 m^2. This allows the users some space to move around while remaining tracked during the test. The experiment software was run on a Windows XP PC with an Intel Core2 Quad Q8200 2.33GHz CPU, 3 GB of RAM, and an nVidia GTX 285 graphics adapter. The HMD and two wireless presenter mice were fitted with infrared markers to allow them to be tracked wrt. both position and orientation. The mice acted as the input devices for the experiment. One of the authors using the experimental equipment is shown in Figure A.4.

Figure A.4: One of the authors wearing the HMD used while holding the two standard wireless presenter mice that served as input devices. Both HMD and mice are fitted with infrared markers. The red lights in the background are coming from the OptiTrack cameras.

The experiment was designed as a within-subjects experiment, where each participant was exposed to all 8 combinations of the levels of the experimental factors. In order to get more data from each participant, each subject was asked to produce five boxes using each technique, i.e. 40 boxes in total. This number

[3]nVisor SX111: http://www.nvisinc.com/product2009.php?id=48
[4]OptiTrack: http://www.naturalpoint.com/optitrack/

of trials was based on a pilot study, which showed that this could be achieved in around one hour per participant. The sequence of 8 techniques was randomized for each participant to filter out any bias that might be caused the by the possible habituation formed by a fixed test sequence.

Before commencing with the actual experiment, the participants would be asked a few demographic questions such as age and gender, and about their previous experience with 3DUI on a scale from 1(=novice) to 5(=expert). Furthermore, the participants were asked about their dominant hand in order to map the controls correctly for the HOC techniques. Subjects were also informed that they could request a break, or termination of the experiment, at any time.

In the experiment, the participants were prensented with a series of yellow wireframe renderings of random boxes, called targets. The targets would always appear at the centre of the tracking volume at approximately chest height of the participant, in order to accommodate maximum mobility of the users around the target. All targets were constrained in size in order to disallow very small, large, and oddly shaped boxes, which would be impossible for the users to recreate. For each presented target box, the users had to attempt to dock a green user controlled wireframe box to match the target box as accurately as possible. The error defined by the distance between the two boxes (see section 3.3) and the completion time were logged for each trial. The virtual test environment presented to the user showed a room with textured checkerboard patterns on the walls. The virtual test environment is illustrated in Figure A.1.

Before doing the 5 trials on a new technique, the participants were given a brief instruction about the technique, and a couple of practice boxes to familiarize themselves with the technique and its mappings to the input devices. After completing all trials, the subjects were informally debriefed.

5 Results

5.1 Participants

A total of 11 people, 10 male and 1 female, participated in the experiment. However, because of the 5-replicate within subjects design, these 11 participants produced a total of 440 boxes to be used in the statistical analysis of the data. All participants were recruited among the local university staff and students. No payment was offered. The mean age of subjects was 30.09 years (σ=8.04 years). The median 3DUI experience level was 2.

5.2 Preliminary Data Analysis

All statistical analysis of results have been made using the free statistics package, R [14]. In all statistical tests, a significance level of $\alpha = 0.05$ has been used.

First of all, the data set was cleared of outliers both with respect to time and precision. These outliers were typically caused by participants who clicked and held a button, causing multiple undesired clicks to be registered by the software. Another type of outlier was the case where people for some reason have spent excessive amounts of time on a trial, typically due to standing in a bad spot with

respect to the tracking system. It was chosen to consider any box error in excess of 1 meter an outlier, and any single trial taking less than 3 seconds or more than 2 minutes an outlier.

Clearing the data set of outliers implies that the experimental data needs to be rebalanced. I.e. there should be an equal number of trials for each of the 8 techniques. The rebalancing was performed by randomly removing samples from each technique. The alternative solution to this would be to add simulated data values based on the existing data. That approach was not attempted. The removal of data samples reduced the total number of trials for each technique from 55 to 43.

Although the design of the experiment ensures that it can be analyzed by analysis of variance (ANOVA) methods, a necessary step is to validate that the assumptions of ANOVA are true with the given data set. To simplify the analysis and presentation of the test data, each combination of technique family and DoF allocation has been assigned a unique ID, which can be seen in Table A.2.

Table A.2: The allocation of unique ID numbers to each technique in relation to the two experimental factors.

Family	DoF allocation			
	3+3+3	3+6	6+3	9
3C	1	2	3	4
HOC	5	6	7	8

The next step of preliminary data analysis was the inspection of two box plots of the data. One plot regarding the precision and another regarding the completion time, both sorted by technique. The box plot regarding precision (box error) is shown in Figure A.5. The completion time box plot is not shown, but both plots feature the same types of problems. As can be seen from the plot, the variance is unlikely to be equal. Especially the box error variance for technique 4 (9DoF-3C) is large. A subsequent Bartlett's test of homogeneous variances reveals that the ANOVA assumption of homogeneous variances is, indeed, not fulfilled, neither for box error ($p \ll 0.001$), nor for the completion time ($p = 0.021$). Furthermore, a Q-Q plot of the residuals of the box error also reveals that they are far from a normal distribution, contrary to another ANOVA assumption. This is also shown in Figure A.5.

To fix this problem, a variance stabilizing transformation was applied. The specific transforms used both belong to the power transformation family, as outlined in [12]. Analysis of the available data, suggests to use the following transforms for the box error, δ, and the time, t:

$$T(\delta) = \frac{1}{\sqrt{\delta}} \ , \ U(t) = \log(t) \tag{A.6}$$

The results of stabilizing the variance (for the box error) can be seen in the right half of Figure A.5. A Bartlett test on the transformed data now accepts

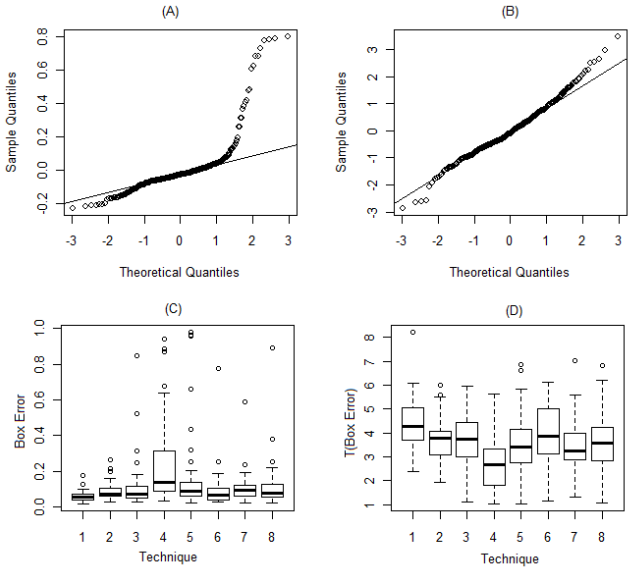

Figure A.5: (A) A Q-Q plot of the box error residuals (circles). The non-linear tendency reveals that the residuals are not normally distributed. (B) A Q-Q plot of the box error residuals after using the variance stabilizing transform. The residuals now fit more nicely on a line. (C) A box plot of box error versus technique used. The plot shows that the variation within each technique appears different, rendering the data problematic for ANOVA. (D) A box plot of the transformed box error versus technique. The variations are now much more equal. The vertical axis is now a transformed version of the box error, implying that smaller values indicate larger errors.

the hypothesis that the variances are equal for both transformed box error ($p = 0.1925$) and transformed time ($p = 0.1132$). It is important to note that the transform imposed on the box error changes it from being a measure of error to a score-like value, i.e. higher values imply more precise dockings. All conclusions made during the subsequent statistical analysis apply in principle only to the transformed variables. However, the transforms used are both bijections over the domains of interest, so they can be unambiguously inverted, making it very likely that the same conclusions are valid for the original variables.

5.3 Test of Hypotheses

To test the hypotheses on the processed data set, a data model taking into account both the two main effects, technique family and DoF allocation, as well as the interaction between the two is set up. The model also incorporates the test subjects crossed with each technique as a random factor to compensate for the within subjects design of the experiment.

5.3.1 Hypothesis 1

The first hypothesis concerned the two families of techniques, 3C and HOC. It hypothesized that 3C techniques would offer superior precision to the HOC techniques. The ANOVA test indicates that there is a significant interaction between the two main factors (F(3,70) = 11.6915, $p < 0.0001$), but no significant difference between the families as such (F(1,70) = 0.001, p = 0.9748). This indicates that some specific techniques perform significantly better or worse than the others. To find out which ones, a post-hoc test was made using Tukey's honest significant difference (HSD) test. This test indicates that technique 1 (3+3+3DoF-3C) is significantly more precise than techniques 3, 4, 5, 7, and 8 with p-values ranging from $p < 0.001$ to $p = 0.044$. Conversely, technique 4 (9DoF-3C) performs significantly worse than all other techniques, p-values ranging from $p < 0.001$ to $p = 0.028$. All of the signicant p-values are presented in Table A.3.

The conclusion wrt. hypothesis 1 must be that it is not true in general. However, our 3C technique with 3+3+3 DoF significantly outperforms most other techniques. Similarly technique 4, which also belongs to the 3C family performs significantly worse than all other techniques. There were no internal differences in the HOC family, meaning that the allocation of DoF for a HOC technique does not matter wrt. docking precision.

Technique 4, the only completely synchronous and symmetric technique, where the user's head also played a role, appears to be worse than all other techniques. For this reason, another post-hoc analysis has been made, where the two 9 DoF techniques are omitted from the data set, to see if any difference among the two families becomes obscured by the bad performance of technique 4. If this is done, then the family of techniques does make a significant difference (F(1,50) = 7.8128, $p = 0.007$), implying that if the definition of 9 simultaneous DoF is excluded, our 3C techniques are more precise than HOC techniques.

Table A.3: p-values from Tukey's honest significant difference test on the precision of the techniques.

Techniques	p-value	Techniques	p-value
1↔3	0.044	2↔4	< 0.001
1↔4	< 0.001	3↔4	< 0.001
1↔5	0.002	5↔4	< 0.019
1↔7	0.001	6↔4	< 0.001
1↔8	0.030	7↔4	0.028
		8↔4	< 0.001

5.3.2 Hypothesis 2

Hypothesis 2 concerned the completion times for boxes. Specifically it stated that 3C techniques would be faster to use, because they featured symmetry between the hands in various ways. Using an ANOVA test reveals a significant interaction

between family and DoF allocation wrt. time (F(3,70) = 4.212, $p = 0.009$), but no significant difference between the families as such. Once again this means that some specific techniques are faster to use than others. A Tukey HSD test was made as post-hoc test to identify these techniques. The results here indicate that technique 3 is significantly faster than techniques 2 ($p = 0.048$) and 5 ($p < 0.001$), and that technique 8 is significantly faster than technique 5 ($p = 0.013$).

Since the fast techniques, 3 and 8, belong to different families, hypothesis 2 cannot be accepted. This means that the family of techniques itself, and hence the symmetry of the hands' actions, appears to have no influence on speed, contrary to the results of [18]. Another interesting result is that splitting a HOC technique into 3+3+3 DoF appears to be a slow box shaping procedure.

5.3.3 Hypothesis 3

Hypothesis 3 stated that the number of DoF controlled simultaneously would affect the precision of the docking such that more DoF controlled at the same time means greater errors in the docking. The same ANOVA analysis that was made for hypothesis 1 was repeated, but this time the point of interest would be the DoF allocation factor. The results show that even in the presence of a significant interaction between DoF and family of techniques, the DoF parameter is still significant (F(3,70) = 8.6285, $p = 0.0001$). A Tukey HSD post-hoc test reveals that defining 9 simultaneous DoF is very significantly less precise than anything else (all $p < 0.001$). Also it shows that using 3+3+3 DoF is significantly more precise than anything else (3+3+3↔3+6 has $p = 0.024$ and (3+3+3↔6+3 has $p = 0.012$).

Thus, hypothesis 3 is accepted in the sense that for precision 9 DoF dockings, 9 simultaneous DoF always performs badly, and 3+3+3 DoF is preferable to any of the other tested options. Using 3+6 or 6+3 DoF does not matter wrt. precision.

6 Conclusion, Discussion, and Perspectives

On the technical side of matters, there are two main contributions of this paper. First, there is the new family of 3C representations and techniques along with the algorithms required to compute the missing corners of a 3C box. The other technical contribution is the box error metric, which can be used to unambiguously compute a distance between two arbitrary boxes. This metric is relevant in any situation, where a quantitative measure of box differences is needed.

For the user-related side of the study, the main contribution is the following list of conclusions about 9 DoF docking precision:

1. The number of DoF controlled simultaneously should be as low as possible. In this experiment the lowest number of simulatneous DoF tested was 3 (three batches of 3 DoF).

2. Controlling 9 simultaneous DoF produces inaccurate dockings compared to all other DoF allocations.

3. Of the tested techniques, placing three corners on a box, one at a time, is the most precise way of docking an arbitrary box into another.

If one tries to extract the general lessons learned from the list above, the result is the following: Good 9 DoF docking precision can be achieved if users are allowed to focus on small subsets of the total DoF and are allowed to lock each subset separately. If DoF are locked in managable batches, it is preferable for users to only operate with positional data rather than explicit scaling and rotation operations.

The 3C techniques proposed in this paper fulfill both of these requirements to varying degrees, which is probably why the 3+3+3 DoF 3C technique emerged as the most precise technique in the experiment. The 3+3+3 3C technique also features the strength that the corners of the created box does not have to be within simultaneous reach of the user's arms, allowing for construction of larger boxes. Furthermore, 3C techniques require no orientation tracking to work.

It is interesting to speculate about the reasons for these conclusions. However, such speculation requires further work to be verified. First of all, people are not used to shaping arbitrary boxes out of thin air in the real world. This means that some level of training may be required if precise control of the 9 simultaneous DoF is to be gained. Furthermore, focusing on a subset of the task not only allows users to put their full attention in one place. It also allows for some period of rest for the arm currently not being used. Therefore, inaccuracy caused by fatigue and arm strain are probably less influential in 3+3+3 DoF than 9 DoF techniques. For this reason, strain-type factors may be interesting to include in a future study.

There are several possible reasons why explicit rotations are more difficult to define than indication of positions. The anatomy of the human wrist limits the amount of rotation possible without resorting to using the shoulder and elbow joints. This is possibly another source of strain and subsequent inaccuracy in dockings. Furthermore, a general observation during this study has been that users generally do not decouple rotation from scaling. Instead, they tend to verify that the orientation is correct by simultaneously matching the scale of the desired box. Indeed, it becomes increasingly difficult to visually determine the orientation of a box as it gets smaller. Another possible limiting factor wrt. rotation accuracy is the devices used. None of the two mouse devices used were shaped like boxes. This means that users may have a hard time relating the shape characteristics of the devices to the orientation of the visualized box.

With respect to completion times, no general conclusions can be made from this study. The two fastest techniques were widely different. One was a 3C technique that featured some degree of symmetry and synchronicity, whereas the other one was a HOC technique, and hence asymmetric. However, the completion times of most of the tested techniques were not significantly different from each other.

Acknowledgements

The authors wish to thank all people who participated in this study. The authors also wish to thank Fonden Vision Nord, who made a generous donation for acquiring some of the central pieces of equipment used in this study.

References

[1] Autodesk, Inc. *Autodesk 3ds Max 2011: Questions and Answers*, 2010.

[2] H. Benko and S. Feiner. Balloon selection: A multi-finger technique for accurate low-fatigue 3d selection. In *3D User Interfaces, 2007. 3DUI '07. IEEE Symposium on*, mar. 2007.

[3] D. A. Bowman, E. Kruijff, J. J. LaViola, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley, first edition, 2005.

[4] W. Buxton and B. Myers. A study in two-handed input. *SIGCHI Bull.*, 17(4):321–326, 1986.

[5] C. Ericson. *Real-Time Collision Detection*. Morgan Kaufmann Publishers, first edition, 2005.

[6] Y. Guiard. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of Motor Behavior*, 19:486–517, 1987.

[7] K. Hinckley, J. Tullio, R. Pausch, D. Proffitt, and N. Kassell. Usability analysis of 3d rotation techniques. In *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*, pages 1–10, New York, NY, USA, 1997. ACM.

[8] A. Huckauf, A. Speed, A. Kunert, J. Hochstrate, and B. Fröhlich. Evaluation of 12-dof input devices for navigation and manipulation in virtual environments. In M. F. Costabile and F. Paternò, editors, *Human-Computer Interaction - INTERACT 2005*, pages 601–614. Springer Berlin / Heidelberg, 2005.

[9] C. Latulipe, S. Mann, C. S. Kaplan, and C. L. A. Clarke. symspline: symmetric two-handed spline manipulation. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 349–358, New York, NY, USA, 2006. ACM.

[10] J. F. Lucas. Design and evaluation of 3d multiple object selection techniques. Master's thesis, Virginia Polytechnic Institute and State University, 2005.

[11] M. R. Masliah and P. Milgram. Measuring the allocation of control in a 6 degree-of-freedom docking experiment. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 25–32, New York, NY, USA, 2000. ACM.

[12] D. C. Montgomery. *Design and Analysis of Experiments*. J. Wiley, seventh edition, 2008.

[13] I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa. The go-go interaction technique: non-linear mapping for direct manipulation in vr. In *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 79–80. ACM, 1996.

[14] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. ISBN 3-900051-07-0.

[15] S. A. Robertson. *Polytopes and symmetry*. Cambridge University Press, Cambridge [Cambridgeshire] ; New York :, 1984.

[16] A. Ulinski, Z. Wartell, and L. F. Hodges. Bimanual task division preferences for volume selection. In *VRST '07: Proceedings of the 2007 ACM symposium on Virtual reality software and technology*, pages 217–218, New York, NY, USA, 2007. ACM.

[17] A. Ulinski, C. Zanbaka, Z. Wartell, P. Goolkasian, and L. Hodges. Two handed selection techniques for volumetric data. *3D User Interfaces, 2007. 3DUI '07. IEEE Symposium on*, mar. 2007.

[18] A. C. Ulinski. *Taxonomy and experimental evaluation of two-handed selection techniques for volumetric data*. PhD thesis, University of North Carolina at Charlotte, Charlotte, NC, USA, 2008. Adviser-Hodges, Larry F.

[19] M. Veit, A. Capobianco, and D. Bechmann. Consequence of two-handed manipulation on speed, precision and perception on spatial input task in 3d modelling applications. *j-jucs*, 14(19):3174–3187, 2008.

[20] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, second edition, 2004.

[21] S. Zhai. *Human Performance in Six Degree of Freedom Input Control*. PhD thesis, University of Toronto, Toronto, Canada, 1995.

[22] S. Zhai and P. Milgram. Quantifying coordination in multiple dof movement and its application to evaluating 6 dof input devices. In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 320–327. ACM Press/Addison-Wesley Publishing Co., 1998.

Paper B

Shape Perception in 3-D Scatterplots Using Constant Visual Angle Glyphs

Rasmus Stenholt and Claus B. Madsen

This paper was submitted as a long paper to the IEEE VR 2012.

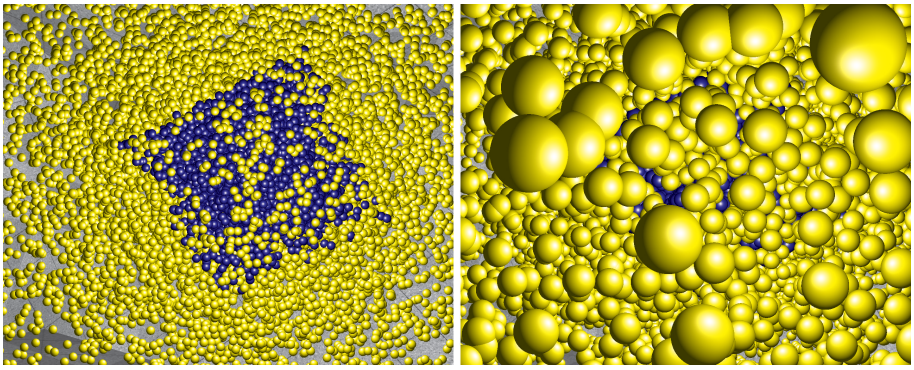

Figure B.1: Two versions of the same 3-D scatterplot viewed from the same position and orientation. (*Left*) Using our proposed constant visual angle (CVA) glyphs. (*Right*) Using traditional perspective glyphs. The blue, box-shaped pattern appears through the yellow noise glyphs with CVA glyphs enabled while remaining hidden and imperceptible using traditional glyph rendering.

Abstract

Visual exploration of clouds of data points is an important application of virtual environments. The common goal of this activity is to use the strengths of human perception to identify interesting patterns in data, which are undetectable using traditional, computational analysis methods. In this paper we seek to identify some of the parameters that affect how well patterns in visualized data clouds can be identified by a human observer. Two of the primary factors tested are the volumetric densities of the visualized patterns and the presence/absence of clutter, or noise, around the displayed patterns. Furthermore, we introduce a new approach to glyph visualization, called constant visual angle (CVA) glyphs, which has the potential to mitigate the effect of noise glyphs at the cost of dispensing with a common real-world depth cue. In a controlled experiment, where test subjects had to locate and select visualized patterns in an immersive virtual environment, we identified several significant results. One is that CVA glyphs ease perception of patterns in noisy environments, while not deteriorating it when noise is absent. Another is the existence of threshold densities, above which pattern identification becomes easier and more precise.

1 Introduction

Visualization of abstract data in order to make sense of it to humans is an old idea.

In [26], the discipline is regarded as having originated in the late 18th century. Much more recently, the use of computers and advances in computer graphics and virtual reality technology has allowed visualization forms that were previously completely infeasible. Today, it is possible to view data visualizations from any conceivable angle, using any desired projection of the data, all in real-time.

The real-time nature of computer visualizations also opens the possibility of interacting with the data. Examples of possible interactions are marking regions of interest, annotating data, or masking out undesired parts of the visualization. Including a human in the process of data analysis comes from the idea that the strong pattern recognition skills of human visual perception can be combined with expert knowledge of the domain from which the data originated to discover interesting patterns otherwise missed by pure computer analysis. This process is often referred to as exploratory visual data mining.

One common approach to data visualization is to map a subset of the variables of the records in a database to the visual attributes of small objects called *glyphs*. Thus, each observation in the data set becomes a glyph in the visualization. The visual attributes of glyphs can be 3-D position, colour, size, shape, texture, etc. Glyph-based visualization of arbitrary data in real-time virtual environments is e.g. treated in [17]. Although many other types of data visualization exist, such as volumetric rendering based on voxels [14], or surface visualizations based on polygons or splats [34], the issues addressed in this paper are aimed at 3-D glyph-based visualizations.

In a very comprehensive survey of perceptual issues related to visualizations [31], Ware notes that little investigation has been made on the human perception of points and structures in 3-D scatterplots. A search of literature on the subject reveals that although some work has been made since Ware's survey, there still seems to be many unexplored areas. Since a 3-D scatterplot is typically the type of plot produced by a 3-D glyph visualization, it seems very relevant to study some of the parameters that may govern how well a human can perceive structures and patterns in such visualizations.

The presence of undesired glyphs, which we will refer to as *noise*, around a pattern is a condition that must be expected in 3-D scatterplots created from most real-world data sets. Thus, occlusion and clutter caused by such noise glyphs becomes a frequently encountered problem as demonstrated in the right part of Figure B.1. In an attempt to diminish this problem, we introduce and test a new way of rendering glyphs, called constant visual angle (CVA) glyphs. Briefly explained, a CVA glyph is a glyph which when fully visible is rendered such that it always covers the same amount of pixels on the screen, and hence the same visual angle, no matter the distance from the viewer to the glyph. The rationale and technical details of CVA glyphs are explained in section 3.

The three parameters studied in this paper are therefore 1) the volumetric density of the glyphs in the scatterplot, 2) the presence/absence of noise glyphs obscuring the view of a pattern, and 3) the use of CVA glyphs in order to diminish the adverse effects of noise. The study will attempt to characterize the perception of the visualizations both from an objective and subjective perspective. This approach allows us to understand if any subjective, perceptual differences reported by subjects can also be objectively measured.

2 Related works

2.1 Visual data mining

Visual data mining (VDM) is a frequently studied area of scientific visualization. The collection of papers [22], gives a good overview of recent work in this field. The concept of *serendipity*, i.e. the human ability to discover something useful even when not looking specifically for it, plays an important role in exploratory VDM. In [10], the concept is explained in a VDM context. Interestingly, it is noted that the concept of *perceptual grouping* is important in VDM. Perceptual grouping is the tendency of human perception to group similar items into larger groups and focus on the group rather than its individual, constituent objects. Thus, a glyph visualization method that actively supports and reinforces this perceptual ability should have an advantage over one that does not do so. This observation is very influential in the design of CVA glyphs. Perceptual grouping is treated in-depth in [25]. The ability of perceptual grouping is closely related to the well-established gestalt laws introduced in the early 20th century [13]. A more recent review of gestalt laws is found in [31].

In [10], some guidelines for visual parameters that can be associated with glyphs are also given. A distinction between static and dynamic attributes is made, static attributes being those that do not vary with time. Dynamic attributes, on the other hand, are those that do vary with time, thus constituting different modes of motion, or animation, of the glyphs.

In [18, 17], the authors describe the implementation of a framework for exploring 3-D scatterplots in immersive virtual environments using both static and dynamic glyph attributes. However, they do not seem to have considered the possibility of improving upon the perceptual grouping aspect by changing the rendering method of the glyphs. The main design consideration wrt. rendering of glyphs in these works seem to be performance-related, in that only tetrahedral glyphs are used, i.e. the simplest possible, closed 3-D polygon surfaces.

Other approaches to glyph-based visualization than those based on small 3-D objects of fixed shape in a scatterplot have been studied in [9]. In that paper, a method for rendering glyphs as a continuous 2-D surface is presented. With this type of glyph rendering, each glyph becomes a tile on a 2-D grid, where the properties of the glyphs are represented as surface curvature, colour and shape.

Mazeika et al, [15], present a method of turning the glyphs in a 3-D scatterplot into a plot of continuous iso-density surfaces. The rationale of doing this is to increase the perception of large-scale structures, i.e. patterns, in the data at the cost of removing the individual observations. In VDM terms, this means that the computer makes the perceptual grouping analysis which would normally be the human observer's task. As will be explained in section 3, CVA glyphs have the potential to offer a view of both overall structures, and indvidual observations, depending on the user's movements.

In a comparative study of user performance in different types of virtual environments [20], the results indicate that typical VDM tasks such as recognizing different glyph shapes and regions of high glyph density are more difficult using head-mounted displays (HMD) than in fish-tank VR type displays. The authors

speculate that this is caused by a lack of overview and a consequently higher mental load when using the HMD. This indicates that the tasks performed by our participants, who also use a HMD, may be very difficult if no support through overview is given. However, as mentioned above, the fact that CVA has the potential provide both structure and details may mitigate this effect. No comparison with other display systems will be made, however.

In [30], a user study based on glyphs rendered using point sprites is presented. This is a technique which can also be used to render CVA glyphs. The authors, however, seem to have completely overlooked the potential perceptual benefits of such glyphs. Instead, they remove the CVA aspect of the glyph rendering in order to make a more realistic visualization. The main focus of the study is on the ability of users to identify and point out maxima and minima in a volumetric visualization of regularly grid-sampled temperature distributions, depending on such parameters as glyph size and density, and background environment. Their motivations for using point sprites is not stated, but it is probably a pure computer performance consideration, such that more glyphs can be rendered at the same computational cost. In the experiment described in [30], the users viewed the visualization in 3-D stereo on a projection screen without the ability to change their viewpoint much. I.e. it was impossible for the user to move into or away from the cloud of glyphs, which is exactly the mode of navigation that reveals the benefits of CVA glyphs. For these reasons, we claim that CVA glyphs are novel, and that their VDM potential prior to the study prensented in this paper is unexplored.

2.2 Dealing with noise in 3-D data visualizations

Currently, there are several main strategies for dealing with glyphs that obscure the view of a pattern of interest in a 3-D visualization.

The first one is to simply remove those glyphs which are unwanted from the visualization. This is typically done interactively, e.g. by letting the user place additional clipping planes. The clipping plane approach is e.g. used in [30, 35]. If the user-controlled clipping planes form a closed volume in space, a 3-D region of interest is created which is sometimes called a 3-D magic lens [29]. User-controlled clipping planes are frequently used with medical data sets to see a specific slice of a data set, e.g. in an MRI scan.

Another approach to dealing with noise is to make the glyphs semi-transparent. This approach has the advantage of both showing the noise and the pattern together. However, it diminishes the ability to assign visual attributes to the surface of the glyphs such as texture and shape. Furthermore, if a perceived pattern is related to the colours of the glyphs, the colours experienced by the user will be a mixture of the noise and pattern colours, which may possibly be confusing. The rendering method known as splatting relies heavily on the use of semitransparency to render discrete point sets as continuous-looking surfaces [32, 36].

A third approach called distortion mapping [4], makes a user-controlled geometric distortion of the 3-D space, which pushes away glyphs far from the user's hand, while condensing those close to the hand.

As will be explained in section 3, CVA glyphs offer a different way of mitigating the adverse effects of noise glyphs.

2.3 Interactions with 3-D scatterplots

The common interactions with a 3-D scatterplot, like the interactions in most other applications of virtual environments, fit into the categories of navigation, selection, manipulation, or system control [2]. There are, however, a couple of specialized versions of selection tasks that are frequently made in VDM applications: 1) Brushing-and-linking, and 2) Multiple object selection (MOS). A prerequisite for both of these important interactions in VDM contexts is the ability to recognize patterns.

Brushing-and-linking is the activity of selecting and highlighting (called *brushing*) a subset of the glyphs in the current projection of the data, in order to see what this highlighted subset looks like under different projections of the data, thereby *linking* an area of interest across several plots of the same data set. The technique was introduced in [3]. A more recent example of application of brushing-and-linking is presented in [12].

Whereas selection tasks traditionally deals with indicating a single object of interest in a virtual world, multiple object selection, or volumetric selection, deals with the same task to be performed on a set of objects. This is an important task, which e.g. enables interactions such as brushing-and-linking as described above. The task given to the subjects in the experiment documented in this paper amounts to a MOS-type task.

In a study by Ulinski et al [27] with experimental factors very similar to the ones investigated in this paper, splatting was used instead of CVA for rendering the glyphs. Ulinski's study showed that the MOS performance of the participants was adversely affected by the presence of occluding noise splats. Furthermore, the results showed that correct selection was more difficult when the splats were sparsely spread out in space instead of being densely clustered. No quantitative interpretation of the terms sparse and dense was given, nor were any intermediate density levels tested.

In a volumetric selection task, where users had to select an entire 3-D object with all its associated polygons [33], Zhai demonstrated that using a box-shaped, semi-transparent selection volume named a silk cursor significantly increased selection accuracy compared to just having a wireframe version of the same selection box. For this reason, semi-transparency will be used for the selection volume used in the experiments of this paper.

3 Theory of constant visual angle glyphs

3.1 Definition

A central contribution of this paper is the introduction of the constant visual angle (CVA) glyph. A constant visual angle glyph is a 3-D glyph like any other, except that its projected size does not change as the viewpoint moves closer or further away. I.e. the perspective foreshortening experienced in real life does not

affect the projected size of CVA glyphs. The immediate consequence of this can be expressed in two equivalent ways:

1. CVA glyphs have a constant screen space size measured in pixels, independent of the viewing distance.
2. CVA glyphs cover an approximately constant solid angle of the viewer's field-of-view, independent of the viewing distance.

Both of these immediate consequences are only true if a glyph is not occluded or outside of the field of view, in which case the glyph will be partially, or entirely, invisible, thus reducing its pixel coverage/visual angle. Also, the visual angle is only constant if viewed at sufficiently small fields-of-view (FoV). I.e. if viewed at the periphery of a very large perspective FoV, then the covered visual angle will be significantly smaller than at the centre of the FoV.

This definition of CVA glyphs has several implications. Some of these are related to the human perception of a visualization of CVA glyphs. Other implications concern the technical implementation of CVA glyph visualizations. These two areas of consequences are treated in the two following subsections.

3.2 Perceptual implications of CVA

When perceiving the real world, we are used to the effect of the visual angle of objects increasing as we move in closer, and vice versa as we move further away. This effect is physically caused by the perspective transformation produced by the lens and retina of the human eye. However, since real-world objects generally do not change their physical size as we change our distance to them, the perceptual interpretation of this phenomenon is that an object's perceived size is constant, although its visual angle varies depending on the viewpoint. This fact is known as the size-distance invariance hypothesis [1], a relation also expressed in Emmert's law [7]. The fact that perceived size is generally constant means that the visual angle covered by an object can be used to judge the distance to the object. This is often known as the *relative size*, or size gradient, depth cue.

With CVA glyphs, the visual angle remains constant, and hence the relative size depth cue will not work. Instead, the most likely inference about a CVA glyph's perceived size is that it must be changing as the viewer moves, to accommodate its visual angle remaining constant. I.e. as you move closer to a CVA glyph, it will appear to become smaller, and vice versa as you move further away. This is shown in Figure B.2. As such, CVA glyphs can be predicted not to conform to the size-distance invariance hypothesis.

Since relative size is no longer a meaningful depth cue with CVA glyphs, it is interesting to investigate how other depth cues are affected. Necessarily, some depth cues should remain unaffected for users to be able to make any sense of the 3-D structure of a CVA glyph visualization. In a VDM context, it is worth noting that size can still be used as a meaningful glyph attribute. This is true because CVA glyphs can be assigned different, but constant, visual angles depending on the data variable mapped to the size attribute. E.g. some glyphs can be chosen to always cover 2° and others 1° in the same visualization.

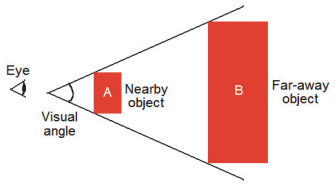

Figure B.2: The consequence of constant visual angle objects. The two objects, A and B, cover the same visual angle of the viewer. However, A and B are placed at different distances from the viewer. If the viewer is able to perceive this depth difference due to other depth cues than relative size, the logical consequence must be that A is smaller than B, as shown in the figure. Furthermore, if the viewer moves closer to A, and the visual angle of A remains constant, then the physical size of A must be decreasing. Conversely, A's physical size will be perceived to be increasing, if the viewer moves away from A, and the visual angle remains constant.

3.2.1 The linear perspective depth cue

One important observation to make about CVA glyphs is that the CVA feature only affects *individual glyphs*. As such, the position of each glyph in a cloud can still undergo the usual perspective transformation. This has two important consequences:

1. The linear perspective depth cue still works on the overall structure of a cloud of CVA glyphs. Vanishing points will still exist in the visualization, and the overall structures of the cloud will be transformed as humans are used to from everyday life. I.e. far-away glyph clouds will cover less screen space than nearby clouds.

2. From a VDM point of view, 3-D position is still a meaningful visual attribute to assign data variables to.

When the familiar linear perspective depth cue is combined with CVA, it provides a couple of potentially very useful viewing properties:

The noise-spreading property: As a viewer moves closer to a CVA glyph cloud, the positions of the glyphs will spread out in the field-of-view due to linear perspective. However, the individual glyph's pixel coverage will *not* increase due to the CVA feature. Thus, nearby CVA glyphs will not begin to fill out the entire field-of-view, obscuring the view of whatever is behind them. The problem of not having CVA glyphs was illustrated in Figure B.1.

This means that any pattern, which is currently obscured by noise glyphs can be cleaned of the noise, by simply moving in closer. The viewer will always be able to select some viewing distance, where the noise glyphs between the viewer and the pattern have spread out so much that the pattern displayed behind them appears relatively noise-free. This property is illustrated in parts A and B of Figure B.3.

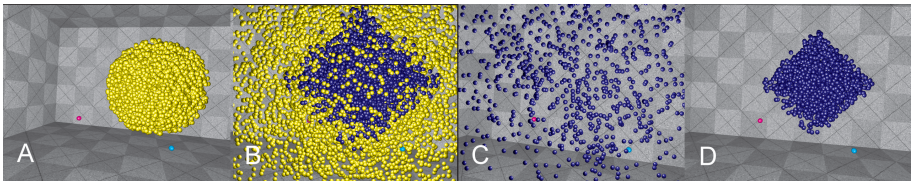

Figure B.3: A CVA glyph cloud with the noise-spreading and pattern-solidification properties illustrated. Yellow glyphs are noise, blue ones are pattern. Note that all glyphs are the same size in pixels across all 4 images. (A) The cloud is viewed at such a large distance that none of the blue pattern is revealed. Instead, the noise is solid. (B) Moving closer to the cloud reveals the blue pattern, relatively unobscured by the noise. This situation also represents a sweet spot, where noise-spreading and pattern-solidification are both helping. (C) Standing inside a pattern causes it to spread out and become imperceptible. (D) Moving away from the pattern causes it to solidify into a recognizable shape.

The main caveat is that the noise and the pattern have to cover more or less disjoint subsets of 3-D space, i.e. some boundary between noise and pattern should exist. If the noise and pattern overlap too much, both will spread out simultaneously.

The pattern-solidification property: If, on the other hand, the viewer is placed in the middle of what is believed to be a pattern of interest, it is possible to move away from the pattern until it appears to solidify in the field-of-view. This property exists because linear perspective will decrease the on-screen distance between glyphs as you move away from them. However, since CVA glyphs retain their pixel coverage, the pattern glyphs will eventually touch and overlap inside the field-of-view, creating the impression of a large, connected structure. Thus, the pattern-solidification property supports the perceptual grouping aspect of human pattern recognition previously mentioned in section 2.1. The solidification property is illustrated in parts C and D of Figure B.3.

Without CVA, the glyphs would simply get smaller and smaller, and not solidify into a pattern. As was the case with the noise-spreading property, it is a requirement that pattern and noise do not overlap too much in 3-D space.

The advantages of the noise-spreading and the pattern-solidification properties can often be combined. This happens if the viewer finds a sweet spot, where the distance to the pattern both causes the noise to spread out and the pattern to solidify at the same time. Such a sweet spot is illustrated in part B of Figure B.3.

3.2.2 Other depth cues

One powerful depth cue which is completely unaffected by the CVA feature is occlusion. I.e. the CVA feature does not prevent nearby glyphs from occluding far away ones. However, correct occlusion depends on the geometry of the CVA glyphs to be well-defined relative to the other glyphs and geometry in the scene. I.e. the constant visual angle must be transformed into a meaningful, physical size in the space of the scene. One approach to achieving this is presented in sec-

tion 3.3. Correct occlusion is likely to be very important for precise interactions with the glyphs in the cloud, since the user can rely on occlusion between glyphs and interaction objects such as 3-D cursors or selection rays.

Depth inferred from shading and cast shadows is also possible with CVA glyphs. As with occlusion, the surface geometry of the glyphs must be well-defined to make correct shading and shadows.

Since correct perspective transformation of the positions of individual glyphs is possible, the depth cue gained from parallax caused by self-motion is also present. According to [31], structure-from-motion is very likely to be a strong cue when perceiving patterns in 3-D scatterplots. This observation is e.g. utilized in spin techniques for desktop viewing of scatterplots, where the plot will continuously spin around some axis [6].

Since both shading and correct perspective transformation of positions in space is possible, depth at short distances can also be inferred from stereoscopic viewing.

In the experiment documented in this paper, the use of a motion-tracked, stereoscopic head-mounted display (HMD) will ensure that the user can take advantage of all of the depth cues mentioned above, with the exception of cast shadows, which was not implemented in the experimental software used for testing.

3.3 Implementation options

There are two options when implementing CVA glyphs in a visualization:

1. Use a polygonal representation as with most other geometry rendered on a GPU.

2. Use point sprites, i.e. images of the glyphs, in the rendering.

For several reasons, the latter option seems much more attractive than the former one.

First of all, the CVA feature comes for free, if point sprites are just naïvely rendered without doing any sort of depth-dependent size correction, e.g. like the correction outlined in [11]. This is true, because a point sprite is simply a fixed-size texture image, which is copied by the rendering pipeline's rasterizer to the screen space positions of the desired set of vertices. I.e. each visible vertex becomes a copy of the sprite image on-screen. Since the pixel-size of the sprite is fixed, glyphs rendered in this way are automatically CVA.

If the same result is to be achieved with polygon-based objects, then each object would have to have a unique depth-dependent scaling transformation applied for each frame to make them always appear with the same screen space size. Moreover, depending on the shape, size, and viewing angle of the glyphs, the parameters of this scaling may only be possible to compute by doing a separate render pass for each glyph to find its current screen space extents. This would be very unattractive from a computational point-of-view.

Using point sprites also has the advantage of only requiring one vertex per glyph to be processed in the vertex and geometry stages of the rendering pipeline.

Using even a simple shape like a tetrahedron would require 4 vertices per glyph to be processed.

The disadvantage of using point-sprites is that the variety glyph shapes which can be used is very limited, provided that the goal is to have the glyphs appear as a true 3-D objects instead of flat 2-D billboards. If the shape of the glyph does not feature full rotational symmetry, then each sprite will require a different view-dependent texture for each frame rendered. This is also quite unattractive from an efficiency point-of-view.

For this reason, spheres are probably the only shape which is currently practical to implement as CVA glyphs. Thus, glyph shape is not a viable visual attribute to use in VDM contexts using CVA glyphs. In fact, this seems to be the main theoretical drawback of CVA glyphs. The only other significant limitation seems to be if users exploring CVA glyphs visualizations do not find the missing relative size depth cue to be acceptable. Investigating this issue is one of the aims of the experiment of this paper.

With spheres, the sprite texture becomes identical across all glyphs and rendered frames. Thus, it becomes very simple to compute the surface geometry of the glyphs needed for correct shading and occlusion. The main issue is to compute the physical radius of each sphere, depending on its current depth. This can be done very simply in a vertex shader, by reversing the formula usually used for approximating the on-screen size of spherical point sprites to appear as if they had undergone normal, perspective transformation. The formula is e.g. presented on p. 277 of [11]:

$$s = 2r \cdot \frac{n}{p_z} \cdot \frac{h}{t-b} \Leftrightarrow r = \frac{s}{2} \cdot \frac{p_z}{n} \cdot \frac{t-b}{h} \tag{B.1}$$

The version of Equation B.1 on the right-hand side is the one of main interest here. r is the physical radius that would currently result from a sphere of pixel diameter s, if the sphere is viewed at a distance of p_z with a vertical screen resolution of h pixels, using a perspective projection with top, bottom and near clipping plane parameters, t, b, and n.

Thus the formula provides the information needed to precisely tell where the sphere boundary is in relation to the sphere centre, and in relation to all other geometry. This information can then be passed to a fragment shader, which takes care of making correct per-pixel shading and depth testing for the glyphs. As mentioned in section 3.2.2, getting this correct is necessary to get the depth-from-shading and depth-from-occlusion cues to work.

One final point to note about the technical implementation of CVA glyphs using points sprites is that it is computationally much more attractive than the alternative of using semi-transparency to see through noise, i.e. splatting. This is due to the fact that semi-transparency implemented using alpha-blending can only be correctly rendered if all glyphs are depth-sorted on a frame-by-frame basis, or if multi-pass techniques or exotic frame buffers, such as A-buffers [5], are employed. Currently no graphics card with an A-buffer exists. No expensive depth-sorting is needed for opaque CVA glyphs.

4 Experiment

4.1 Experimental method & hypotheses

The goals of the experiment can be expressed through some research questions. One goal is to investigate the usefulness of CVA glyphs for pattern recognition purposes. In this context, it is of particular interest to investigate the following questions:

1. How well can people perceive patterns using CVA glyphs as compared to regular glyphs?
2. How does the presence of noise affect user performance with and without CVA glyphs?
3. Under which conditions, if any, are CVA glyphs beneficial to use?

From these questions, a couple of testable hypotheses can be stated, which forms part of the basis of the design of the experiment.

H1 CVA glyphs will provide better pattern perception and pattern reproduction performance than regular glyphs in noisy environments.

H2 CVA glyphs will not pose a problem to the perception and interactions of the users compared to regular glyphs.

Hypothesis 1 is justified by the fact that CVA glyphs have the potential of spreading out noise, as described in section 3.2. Furthermore, if the user can find a sweet spot to view the pattern from, the pattern should solidify to create a relatively unobscured view of a continuous pattern. None of this will happen with regular 3-D glyphs.

The justification of hypothesis 2 is that although CVA glyphs are missing the natural relative size depth cue, the amount of other depth cues present is still large. The pattern-solidification property may also in some cases compensate for the depth perception lost due to the missing relative size cue. This is true because pattern-solidification may help stimulate the perceptual grouping aspect of human perception, which is something that will not happen with regular glyphs.

A secondary objective is to investigate various density levels of the presented glyphs. This is necessary, because the statements made in the first two hypotheses can only be reasonably expected to be true, if there are sufficient glyphs present to form any pattern at all. The work of Ulinski et al [27, 28] supports this as a necessary parameter, since they already identified that there is a significant difference in selection performance in sparse and dense clouds of splats. However, it seems important to re-investigate this result in terms of quantitative density levels. Therefore, the necessary research questions about density must be:

1. Where is the lower density threshold, if any, that makes pattern shape perception very difficult, or even impossible?

2. Where is the upper density threshold, if any, that makes pattern shape perception trivial?
3. How does the use of CVA glyphs interact with these density thresholds, i.e. do the thresholds change if CVA glyphs are used?

These questions can also be rephrased as testable hypotheses:

H3 There exists a lower threshold below which pattern perception becomes difficult and pattern selection performance decreases.

H4 There exists an upper threshold above which pattern perception becomes easy and pattern selection performance increases.

H5 There is statistically significant interaction between density and the use of CVA glyphs.

Hypothesis 3 and 4 can both be justified through the results of the experiments presented in [27], and by observing that the importance of perceptual grouping must mean that to see a pattern, there must at least be enough visualized elements to visually form a group.

The solidification property justifies why hypothesis 5 ought to be correct. The possibility of CVA solidification should mean that patterns can be perceived as continuous surfaces at lower density levels than without CVA.

4.1.1 Experimental factors

From these hypotheses, the factors of the experiment can be identified:

- **Density** of the pattern glyphs, measured in glyphs/m^3. This factor has 5 levels: 1,000 - 5,000 - 10,000 - 50,000 - 100,000. These specific levels were chosen, since they clearly show differences when inspected on a regular screen. The upper limit of 100,000 was chosen, such that the pattern shape is fairly straightforward to see, while still allowing reasonable frame rates (app. 20-25 fps) on the available hardware platform. The lower limit of 1,000 is chosen since it is judged to be very difficult to see any structure to the glyphs at this level. Thus, any upper and lower pattern shape detection thresholds should be located somewhere in the chosen range.
- **Noise** is a two-level factor with the levels 'yes' and 'no', depending on the precense (yes) or absence (no) of noise. The amount of noise glyphs scales proportionally to the density of the pattern, such that there are always 4 times more noise glyphs than pattern glyphs. However, the noise glyphs are always spread out over a larger volume of space around the patterns, effectively producing a noise density of around 20% of the pattern density.
- **CVA** is another two-level factor with the levels 'yes' and 'no' depending on the use of CVA glyphs (yes) or regular glyphs (no). In cases where CVA glyphs are not used, the on-screen radius of the glyphs will change in the natural way, as e.g. described in the left-hand version of Equation B.1.

This implies that the experiment is going to be a 3-way factorial design with $5 \times 2 \times 2 = 20$ unique experimental conditions. In order to collect more data from the participants, the experiment is carried out using a within-subjects design with replication. A small pilot experiment was performed, revealing that 2 replications of each condition was achievable in 1-1.5 hours per participant, which was judged to be reasonable. Having 2 replications means that each participant will go through a total of 40 trials. An overview of all test conditions is summarized in Table B.1.

Table B.1: A table of all test conditions. The numerical entries in the table are the unique ID's assigned to each condition. E.g. condition 4 has a density of 5,000 glyphs/m^3 with CVA glyphs enabled and no noise.

Without noise					
CVA	Density [glyphs/m^3]				
	1,000	5,000	10,000	50,000	100,000
Yes	0	4	8	12	16
No	1	5	9	13	17

With noise					
CVA	Density [glyphs/m^3]				
	1,000	5,000	10,000	50,000	100,000
Yes	2	6	10	14	18
No	3	7	11	15	19

4.1.2 User task & response variables

The task to be carried out by the user is closely related to the desired response variables that we wish to measure. The user task in all trials is as follows. The user is presented with a pattern rendered as blue glyphs. If the scenario also involves noise, the noise glyphs will be yellow. The choice of colours is based on getting good contrast between pattern and noise. The task of the user is then to first explore the glyph cloud to see the shape of the pattern, then to reproduce the shape of the pattern in-situ. This scheme allows for several response variables, objective and subjective, to be measured.

In order to be able to get reliable, quantitative results about the reproduction of a pattern, the pattern shape needs to be one which is both simple to recognize and reproduce. For this reason, we have chosen that the pattern shape in all cases is a random 3-D box featuring all possible 9 degrees-of-freedom (DoF). That is, the pattern boxes are randomly translated, rotated, and scaled. The possible box volumes, however, are constrained to manageable sizes, their dimensions being forced to lie between 25 and 75 cm. Thus, the task of the user will be to create

the perceived, minimal oriented bounding box (OBB) [8] around the blue pattern glyphs. The reasons for this choice are:

1. 3-D boxes are well-known and recognizable to most people.
2. 3-D boxes have enough DoF to be non-trivial to reproduce. I.e. you have to know the locations of the faces and corners of the box to have perceived it correctly, and subsequently be able to reproduce it accurately.
3. Effective and precise 3-D box construction interaction techniques exist, e.g. hand-on-corner (HOC) introduced in [27] or 3-corners (3C) introduced in [23]. The latter is chosen, since it has been tested to be more precise than HOC. Specifically, the technique presented as 3+3+3 DoF 3C is employed. Briefly put, this means that the user creates a box by indicating three corners on the desired box, pointing them out one at a time. See [23] for more details on 3C techniques.
4. It is possible to quantitatively measure the difference between two boxes [23].

An alternative approach to the experiment, would be to include different classes of 3-D pattern shapes, such as spheres, tetrahedrons, cylinders, etc. However, we have chosen not to do so for several reasons. First of all, this would add an additional multi-level factor to the experiment, which would increase the number of unique test conditions to an unmanageable amount for this experiment. Secondly, immersive virtual environment versions of construction techniques for most other shapes do not seem to have been extensively studied in literature, making efficient, standardized production of them a study of its own. Thirdly, one may may conjecture that the findings of an experiment with box patterns may generalize to other shapes - at least to those with a comparable amount of DoF and symmetry.

Since we wish to understand the difficulty of perceiving patterns, both from an objective and a subjective viewpoint, the following response variables were chosen:

- **Box error:** The box error is an objective, metric measurement of the difference between the correct and the perceived, user-created box. The metric is computed as described in [23].
- **Perception difficulty:** This is a subjective measurement. The user is asked how difficult it was to perceive the bounding box of the pattern on a scale from 1 (trivial) to 10 (impossible).
- **Selection difficulty:** Reproducing the box pattern in-situ effectively amounts to a multiple object selection task, where the user selects the glyphs in the pattern using a tightly fitting box. The user is asked how difficult it was to reproduce the box pattern on a subjective 1-10 scale like above.
- **Selection sensitivity:** In selection experiments, the sensitivity is often used as a measure of quality. In this context, the sensitivity is the ratio of the

number of selected pattern glyphs to the total number of pattern glyphs. This is an objective, quantitative measurement. It does not necessarily correlate with the box error, since the whole pattern can be selected by many different boxes. Thus, any deviation from the minimal bounding box, will adversely affect the box error, but not necessarily the sensitivity.

- **Completion time:** The completion time of each trial is also measured as an objective response variable. Even though the hypotheses do not deal explicitly with completion time, it is nonetheless interesting to see if there are any differences between the test conditions wrt. speed.

The noise and pattern glyphs are spatially placed such that the pattern is completely embedded inside a spherical cloud of noise. The spherical shape ensures that no inferences about the shape of the pattern can be made from the shape of the noise. However, the pattern volume must at the same time be completely free of noise glyphs, allowing the user, in principle, to make a perfect selection of pattern glyphs. To help the participants' perception of the volume covered by their currently created box, the colour of pattern glyphs changes from blue to green when selected. Similarly, false positives, i.e. selected noise glyphs, change from yellow to red, once selected. This produces an alternative way of judging the current correctness of the constructed box by maximizing the amount of green glyphs, and minimizing the amount of red ones. An example illustration from the experiment is shown in Figure B.4.

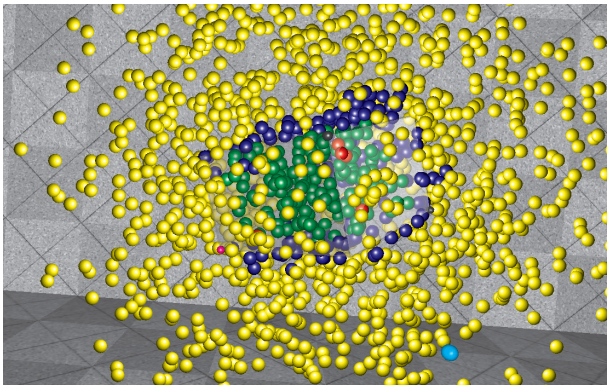

Figure B.4: A screenshot from one of the trials with 10,000 glyphs/m^3 and CVA enabled. The green (pattern) and red (noise) glyphs are those currently selected by the semi-transparent, user-created box. The user controlled pointers are represented by small cyan and magenta spheres.

The final consideration to make is the sampling strategy for the pattern box. The glyph pattern must necessarily have the sampled target box as minimal bounding box. Otherwise, the ground truth to measure the box error against disappears. This means that either 4 glyphs must always be placed at 4 non-coplanar corners of the box, or that some glyphs must be placed on each face of the box.

Figure B.5: The HMD and mice used in the experiment. The HMD and both mice are fitted with infrared reflective markers to be identifiable by the OptiTrack system.

We have chosen the latter approach, since pilot tests revealed that always having 4 corners explicitly represented as glyphs was too conspicuous. It made the box shape identification a trivial task of indicating 3 already visualized locations in space, thus defeating the goal of finding density level thresholds.

The approach is therefore to always sample 3 glyphs on each of the 6 faces of the pattern box. Apart from the 18 face glyphs, all other glyphs are randomly sampled from the entire pattern box' volume. Thus, there are no guarantees that the pattern glyphs will be evenly spread out inside the volume of the box, as could e.g. be achieved using sampling strategies such as n-rooks or Hammersley sampling [24]. The random sampling approach was chosen in order to best approximate the most likely conditions, if the glyph positions had been mapped from an arbitrary real-world data base. In any case, random sampling also represents the worst-case scenario for the purposes of human pattern recognition - any kind of regularity in the positioning of the glyphs, would most likely help in perceiving the correct pattern.

4.2 Experimental equipment

The experiment was performed using an nVisor SX111 HMD and a 24-camera OptiTrack setup for motion tracking. The HMD features a 66% partial overlap field-of-view of 102° (horizontal) by 64° (vertical). The HMD resolution is 1280×1024 pixels per eye. The setup of the tracking system allows the user to move wihtin a 2.25 m radius from the centre of the tracked space. Two off-the-shelf wireless presenter mice were fitted with tracking markers and used as interaction devices. All necessary actions, i.e. indications of perceived box corners, were performed by clicking the left mouse buttons. The HMD and mice are shown in Figure B.5.

The experiment application was a custom-made OpenGL renderer running on a 32 bit Windows XP installation with a Intel Q8200 2.33 GHz quad-core CPU and 4 GB of memory. The motion tracking software was based on the NaturalPoint Tracking Tools API. The graphics card used was based on an nVidia GTX 285 GPU.

4.3 Experimental procedure

Before doing any trials of the experiment, each participant answered a few demographical questions about age, gender, and prior 3DUI experience. Then, a brief explanation of the equipment followed to allow the participants to familiarize themselves with the adjustment knobs of the HMD, and the left button position on the mice. All participants were made aware that they could request breaks, or termination of the experiment, at any time. After donning the HMD, the participants were presented with a few easily perceptible practice patterns of high density, in order to explain the procedure of each trial. In connection with this, a short training session in the use of the 3+3+3 DoF 3C box shaping technique was given, where participants could practice replicating some easy box patterns. Participants were also explained about the difference between blue pattern glyphs and yellow noise glyphs, and the colour changes induced by selecting glyphs.

The instruction given to the participants was to approximate the minimal bounding box as well as possible in all cases. The participants were not made aware of the CVA glyphs prior to, or during, the trials. Thus, they remained ignorant of the peculiarities of CVA glyphs, unless they noticed anything by themselves. Participants were allowed to restart individual trials if they felt that they had made a mistake in pointing out a box corner. In cases where the pattern's bounding box was not perceptible, participants were instructed to make their best guess, nonetheless.

After the practice session, the actual experiment with logging of data began. The sequence of test conditions was randomized to avoid any possible bias induced by a fixed sequence. Both replications of the same test condition were in immediate sequence. All target patterns were randomized on a trial-by-trial basis. After the two replications of each condition, participants were asked about their subjective opinion of 1) the difficulty of perceiving the bounding box around the blue glyphs, and 2) the difficulty of reproducing the perceived bounding box, both on the previously mentioned 1-10 scale. As previously explained, each participant tried all 20 test conditions twice. The test conditions are presented in Table B.1. Upon completion of all trials, an informal debriefing took place.

5 Results

5.1 Participants

A total of 19 unpaid participants took part in the test, 17 male and 2 female. However, two of the male participants had to quit the experiment during the training phase due to cyber-sickness. Thus, all further analysis is only based on the 17 participants who completed the entire experiment. In the present case, the 17 participants tried all conditions twice, so the sample size is 34 for all conditions. The collected data set represents 680 completed trials in total. All participants were recruited at the local university campus. The mean age was 28.5 years. The median 3DUI experience level, reported on a subjective 1 (least experienced) to 5 (most experienced) scale was 2. All subsequent statistical analyses were performed using R [21], and a significance level of $\alpha = 0.05$ was used in all tests.

5.2 Pre-analysis

A pre-analysis was performed to see if the data conformed to the requirements of ANOVA analysis. That is, independence, equal variance, and normality. Independence is secured by the fact that the trials were independently performed by different participants, and the fact that the sequence of the test conditions was randomized for each participant. However, Bartlett's and Levene's tests of equal variance reveal that the equal variance assumption is likely violated for all measured response variables. Although some controversy exists on the issue of robustness to violations of this assumption, recent literature on the issue, such as Norman's paper [19], indicates that even quite large violations of ANOVA's assumptions do not lead to the wrong conclusions in general.

Based on the conclusions of [19], the relatively large total sample size of 680 trials, and a comparison of the outputs of a Friedman test (the non-parametric version of ANOVA) and a regular ANOVA, which revealed no noteworthy differences, we chose to go ahead with ANOVA regardless of the violations. All post-hoc tests are made using Tukey's honest significant difference (HSD).

We decided to apply variance stabilizing transformations on each of the response variables as e.g. outlined in [16], in order to improve the situation. However, we could not entirely fix the issue of equal variance for most of the response variables. The selection difficulty did not indicate any need for variance stabilization. The data analysis is based on the transformed responses. The following transformations were estimated from the data and applied to the box errors, δ, the perceptual difficulties, p, the completion times, t, and the sensitivities, s:

$$\delta' = \log(\delta)\ ,\ p' = \sqrt{p}\ ,\ t' = \frac{1}{\sqrt{t}}\ ,\ s' = s^{2.5} \tag{B.2}$$

As a final preliminary analysis, a boxplot of the transformed variables reveals that, at least visually, there appears to be interesting differences among the tested conditions. As an example, in Figure B.6, the transformed box errors and perceptual difficulties are displayed grouped by condition.

5.3 Factors, interactions, and hypotheses

5.3.1 The big picture

A 3-way within-subjects ANOVA with replication analysis of the transformed box error reveals that all factors and interactions between 2 and 3 factors, except for the main effect of using CVA, are significant. The significant p-values here range from 0.032 to $\ll 0.001$. I.e. there are significant differences among the specific test conditions (3-factor interactions) as well as significant overall differences caused by the main effects and various combinations of the main effects.

The results wrt. perception difficulty, selection difficulty, and sensitivity show significant effects and interactions in mostly the same places as the box error.

The completion time shows few significant effects compared to the other response variables. The main significant effect being that trials with noise are significantly slower to complete than those without noise ($p \ll 0.001$).

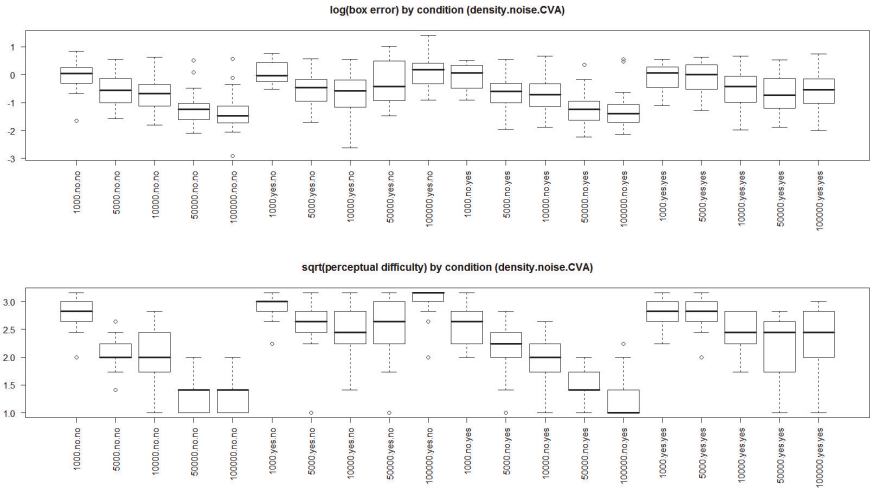

Figure B.6: Boxplots grouped by condition of (top) the transformed box error, which is an objective measure of task difficulty, and of (bottom) the transformed perceptual difficulty, a subjective measure of task difficulty. The leftmost 10 conditions are all without CVA glyphs, whereas the rightmost 10 conditions are all with CVA glyphs. If viewed as 4 chunks of 5 boxes, the conditions within each chunk show the result of changing the density, while keeping the other factors constant. Note that all 4 low density conditions (1,000 glyphs/m^3) appear to both objectively and subjectively difficult. Also note the apparent beneficial difference that CVA glyphs make in the most dense, noisy conditions. Furthermore, it is interesting to see that the participants' subjective perceptions of difficulty are similar to the objectively measured errors.

One main trend of the experiment is that participants found it difficult to perceive the box patterns in the lowest density cases with 1,000 glyphs/m^3. These cases were significantly more difficult than the other tested conditions, except for the noisy, 100,000 glyphs/m^3 case without CVA glyphs, which turned out to be the most difficult tested condition, no matter which response variable is considered.

Another main trend is that the scenarios with the highest densities and no noise were significantly easier than most of the other tested conditions.

The main goal of the experiment was to investigate the 5 proposed hypotheses. This means that instead of trying to understand every significant difference in the data set, we have chosen to only investigate those in detail that matter to the hypotheses.

5.3.2 Hypotheses 1 & 2

H1 stated that the use of CVA glyphs would both help users to perceive and reproduce the displayed patterns more precisely in noisy environments.

Investigating this with respect to the pattern perception first, the main variable to look at is the reported perceptual difficulty. Wrt. perceptual difficulty, CVA in itself is a significant effect (p=0.026). Interestingly, this means that across all tested conditions, the participants actually felt it was easier to perceive patterns with CVA glyphs than without. If we then check how CVA glyphs fare in the most noisy environments, i.e. those with noise and the highest pattern density, we learn that CVA glyphs do indeed make a significant difference (p = 0.021). The visual difference caused by CVA in this situation was the one illustrated in the teaser image on the first page.

If H1 is investigated using the objective box error, we find that CVA makes a significant beneficial difference ($p < 0.001$) in the most dense, noisy condition, but not as a general main effect. Investigating H1 wrt. the reported selection difficulty shows the same result (p = 0.034), and for the sensitivity of the selection, this result is also consistent ($p < 0.001$) .

This means that once the amount of noise becomes very high, effectively covering the view of an otherwise well-defined pattern, CVA glyphs make a significant difference, no matter how we have chosen to measure the difficulty of the task. However, contrary to the statement of H1, CVA does not make a significant difference in general in noisy environments. The correct statement of a revised H1 would therefore be that CVA glyphs help the perception of patterns in noisy environments where occlusion of the pattern becomes dominant.

H2 stated that using CVA glyphs would not cause perceptual problems compared to rendering the glyphs in the natural way.

From the analysis of H1 above, H2 has already proven to be true. Perhaps the most surprising result here is that, in general when using the participants' subjective opinions about the difficulty of perceiving and selecting patterns, CVA glyphs prove significantly better to use than regular glyphs. When measured using the objective box error, there is no significant difference across all scenarios.

This means that CVA glyphs do not reduce people's objective ability to create the correct box around a pattern, and that people subjectively perceive it as being significantly easier with CVA glyphs. Thus, H2 is believed to be true, based on the available data.

5.3.3 Hypotheses 3 & 4

H3 stated that there exists a lower density threshold, below which pattern perception becomes more difficult. To test this, the main factor of interest is density. The performed ANOVA shows that this factor is significant wrt. all measured response variables except for completion time, all p-values $\ll 0.001$.

Digging into the pairwise comparisons made by a Tukey's HSD comparison, shows that the participants find the density level of 1,000 glyphs/m^3 more difficult than all the other density levels, no matter which response variable is used (except for completion time). The p-values here range from 0.005 to < 0.001. The only interpretation of this must be that a lower threshold level (or, more likely, a threshold range) exists somewhere between 1,000 and 5,000 glyphs/m^3, which causes the perception of a shape from a discrete set of glyphs to become much

more difficult, if not impossible. H3 is therefore unambiguously confirmed by the data.

H4 postulates the existence of an upper density threshold, above which pattern perception becomes easier. Investigating the data shows significant differences between 10,000 and 50,000 glyphs/m^3 for the response variables box error, perception difficulty, and sensitivity, p-values in the range 0.044 to < 0.001. This means that the desired threshold probably lies between 10,000 and 50,000 glyphs/m^3.

The only response variables not indicating the threshold to be at the same location, is the self-reported selection difficulty that produces a significant difference between 5,000 and 50,000 glyphs/m^3. Since this range mostly overlaps the range indicated by the other response variables, it seems most reasonable to say that more than 10,000 glyphs/m^3 are needed to significantly improve pattern perception. Thus H4 is found to be in agreement with the data, and hence true.

5.3.4 Hypothesis 5

The final hypothesis, H5, stated that the use of CVA glyphs would change the density thresholds postulated by H3 and H4. To investigate this hypothesis, the interaction term between CVA and density is of interest. This interaction proved significant wrt. all response variables except for completion time.

The box error shows that the threshold between 1,000 and 5,000 glyphs/m^3 vanishes when CVA glyphs are used. Conversely, the significant difference between 10,000 and 50,000 glyphs/m^3 vanishes without CVA glyphs. The broad interpretation of this must be that at low densities, the familiar relative size depth cue is of more help than CVA, but at higher densities CVA provides better pattern perception. I.e. the thresholds have been moved or smoothed out by the use of CVA.

The reported perception difficulty also shows that with CVA glyphs enabled, the significant difference between 1,000 and 5,000 glyphs/m^3 disappears. However, the threshold between 10,000 and 50,000 glyphs/m^3 remains significant. Thus, perceptually the use of CVA glyphs has made the low density cases more similar.

With the subjective selection difficulty, the results of the analysis gives the same conclusion as with the box error. I.e. that CVA glyphs makes it easier to select glyphs at high densities, but makes no difference at low densities. The sensitivity thresholds are unaffected by the use of CVA glyphs.

Thus, the conclusion about H5 must be that the use of CVA glyphs in general has an effect on pattern perception density thresholds.

6 Discussion & Conclusion

In summary, this paper has made the following contributions:

1. The introduction of constant visual angle (CVA) glyphs, a novel approach to 3-D glyph rendering, where glyphs are rendered such that the individual glyphs always appear with the same pixel-coverage on-screen.

2. An analysis of the the perceptual consequences of CVA glyphs based on current theories, in particular why CVA glyphs have the potential to be useful as noise removal technique in visual data mining of 3-D scatterplots.

3. A discussion of various issues related to the technical implementation of CVA glyphs was given, arriving at the conclusion that point sprites is probably the best, currently feasible implementation option.

4. A demonstration through a rigourous experiment carried out in an immersive virtual environment that CVA glyphs do indeed have the expected advantages over regular glyphs, especially in conditions featuring a lot of occlusion from noise glyphs.

5. Identification of possible threshold regions of glyph density which significantly change participants' ability to perceive large shapes and patterns from discrete 3-D glyphs.

The fact that CVA glyphs not only proved usable, but in many cases superior to glyphs rendered in a more natural way, is an interesting result in itself. This means that the loss of a real-world depth cue can be an advantage in some applications. This result is made even more valuable by the fact that the proposed CVA glyphs not only outperformed traditional glyphs in very noisy conditions, but also, in general, did not perform any worse than realistic glyphs in other conditions.

One interesting, but informal, observation made during the course of all the trials of the experiment, is that only 3 out of 17 participants noticed anything unnatural about the CVA glyphs. Furthermore, nobody made any statement that the CVA glyphs were annoying or unpleasant. The participants had no problems in using the noise-spreading and pattern-solidification properties of CVA glyphs, which seemed quite natural to them. CVA glyphs also proved to be advantageous, in that the interactions needed for reproducing the pattern boxes were more manageable in the very noisy cases. This observation, made by several of the participants, is caused by the fact that the noise-spreading property of CVA glyphs also causes the user controlled pointers to be more easily visible through the noise.

Finally, the density thresholds found through the experiment are not only useful with CVA glyphs. The thresholds provide general knowledge that can be used in planning the sampling density of data for any 3-D scatterplot rendering, if the goal is to have the viewers perceive shapes and patterns in the plot.

Acknowledgements

The authors wish to thank Fonden Vision Nord for generously donating the funds needed to acquire some of the equipment used in this study.

References

[1] W. W. Blessing, A. A. Landauer, and M. Coltheart. The effect of false perspective cues on distance and size-judgments: an examination of the invariance hypothesis. *The American journal of psychology*, 80(2):250–256, 1967.

[2] D. A. Bowman, E. Kruijff, J. J. LaViola, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley, first edition, 2005.

[3] A. Buja, J. A. McDonald, J. Michalak, and W. Stuetzle. Interactive data visualization using focusing and linking. In *Proceedings of the 2nd conference on Visualization '91*, VIS '91, pages 156–163, Los Alamitos, CA, USA, 1991. IEEE Computer Society Press.

[4] M. S. T. Carpendale, T. Carpendale, D. J. Cowperthwaite, and F. D. Fracchia. Distortion viewing techniques for 3-dimensional data. In *Proceedings of the 1996 IEEE Symposium on Information Visualization (INFOVIS '96)*, INFOVIS '96, pages 46–, Washington, DC, USA, 1996. IEEE Computer Society.

[5] L. Carpenter. The a -buffer, an antialiased hidden surface method. *SIGGRAPH Comput. Graph.*, 18:103–108, January 1984.

[6] A. W. Donoho, D. L. Donoho, and M. Gasko. Macspin: Dynamic graphics on a desktop computer. *IEEE Comput. Graph. Appl.*, 8:51–58, July 1988.

[7] E. Emmert. Größenverhältnisse der nachbilder. *Klinische Monatsblätter für Augenheilkunde und für augenärztliche Fortbildung*, 19:443–450, 1881.

[8] C. Ericson. *Real-Time Collision Detection*. Morgan Kaufmann Publishers, first edition, 2005.

[9] C. Forsell, S. Seipel, and M. Lind. Surface glyphs for efficient visualization of spatial multivariate data. *Information Visualization*, 5:112–124, June 2006.

[10] E. Granum and P. Musaeus. *Constructing virtual environments for visual explorers*, pages 112–138. Springer-Verlag, London, UK, 2002.

[11] M. Gross and H. Pfister. *Point-based graphics*. Morgan Kaufmann series in computer graphics. Morgan Kaufmann, 2007.

[12] P. Isenberg and D. Fisher. Collaborative Brushing and Linking for Co-located Visual Analytics of Document Collections. *Computer Graphics Forum*, 28(3):1031–1038, June 2009.

[13] K. Koffka. *Principles of Gestalt Psychology*, volume 7. Harcourt, Brace and World, 1935.

[14] A. Kratz, M. Hadwiger, R. Splechtna, A. Fuhrmann, and K. Bï¿½hler. Gpu-based high-quality volume rendering for virtual environments. *in Proceedings of AMI-ARCS*, 2006.

[15] A. Mazeika, M. H. Böhlen, and P. Mylov. Using nested surfaces for visual detection of structures in databases. In *Visual Data Mining*, pages 91–102. 2008.

[16] D. C. Montgomery. *Design and Analysis of Experiments*. J. Wiley, seventh edition, 2008.

[17] H. R. Nagel. *Exploratory Visual Data Mining in Spatio-Temporal Virtual Reality*. PhD thesis, Faculty of Engineering and Science, Aalborg University, 2005.

[18] H. R. Nagel, E. Granum, S. Bovbjerg, and M. Vittrup. Immersive visual data mining: The 3dvdm approach. In *Visual Data Mining*, pages 281–311. 2008.

[19] G. Norman. Likert scales, levels of measurement and the "laws" of statistics. *Advances in health sciences education theory and practice*, 15(5):625–632, 2010.

[20] W. Qi, R. M. Taylor, II, C. G. Healey, and J.-B. Martens. A comparison of immersive hmd, fish tank vr and fish tank with haptics displays for volume visualization. In *Proceedings of the 3rd symposium on Applied perception in graphics and visualization*, APGV '06, pages 51–58, New York, NY, USA, 2006. ACM.

[21] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2011. ISBN 3-900051-07-0.

[22] S. J. Simoff, M. H. Böhlen, and A. Mazeika, editors. *Visual Data Mining - Theory, Techniques and Tools for Visual Analytics*, volume 4404 of *Lecture Notes in Computer Science*. Springer, 2008.

[23] R. Stenholt and C. B. Madsen. Shaping 3-d boxes: A full 9-degree-of-freedom docking experiment. In *Virtual Reality Conference (VR), 2011 IEEE*, pages 103–110, March 2011.

[24] K. Suffern. *Ray Tracing from the Ground Up*. A. K. Peters, Ltd., Natick, MA, USA, 2007.

[25] A. Treisman. Perceptual grouping and attention in visual search for features and objects. *Journal of Experimental Psychology: Human Perception and Performance*, 8:194–214, 1982.

[26] E. R. Tufte. *The Visual Display of Quantitative Information, 2nd edition*. Graphics Press, 2 edition, May 2001.

[27] A. Ulinski, C. Zanbaka, Z. Wartell, P. Goolkasian, and L. Hodges. Two handed selection techniques for volumetric data. *3D User Interfaces, 2007. 3DUI '07. IEEE Symposium on*, mar. 2007.

[28] A. C. Ulinski. *Taxonomy and experimental evaluation of two-handed selection techniques for volumetric data*. PhD thesis, University of North Carolina at Charlotte, Charlotte, NC, USA, 2008. Advisor - Hodges, Larry F.

[29] J. Viega, M. J. Conway, G. Williams, and R. Pausch. 3d magic lenses. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*, UIST '96, pages 51–58, New York, NY, USA, 1996. ACM.

[30] N. Wang, A. Paljic, and P. Fuchs. *A study of Perception of volumetric rendering for immersive scientific visualization*, pages 145—152. 2010.

[31] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, second edition, 2004.

[32] L. A. Westover. Splatting: A parallel, feed-forward volume rendering algorithm. Technical report, Chapel Hill, NC, USA, 1991.

[33] S. Zhai. *Human Performance in Six Degree of Freedom Input Control*. PhD thesis, University of Toronto, Toronto, Canada, 1995.

[34] H. Zhang and A. Kaufman. Interactive Point-based Isosurface Exploration and High-quality Rendering. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):1267–1274, 2006.

[35] E. V. Zudilova and P. M. A. Sloot. Virtual reality and desktop as a combined interaction-visualisation medium for a problem-solving environment. In *Proceedings of the 2003 international conference on Computational science: PartIII*, ICCS'03, pages 1025–1034, Berlin, Heidelberg, 2003. Springer-Verlag.

[36] M. Zwicker, H. Pfister, J. van Baar, and M. Gross. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 371–378, New York, NY, USA, 2001. ACM.

Paper C

Brush, Lasso, or Magic Wand? Picking the Right Tool for Large-Scale Multiple Object Selection Tasks

Rasmus Stenholt and Claus B. Madsen

This paper was submitted as a full paper to the IEEE 3DUI 2012.

Abstract

The task of multiple object selection (MOS) in immersive virtual environments is important and still quite unexplored. The difficulty of efficient MOS increases with the number of objects to be selected. E.g. in small-scale MOS, only a few objects need to be simultaneously selected. This may be accomplished by serializing existing single-object selection techniques. In this paper, we explore various MOS tools for large-scale MOS. That is, when the number of objects to be selected are counted in hundreds, or even thousands. This makes serialization of single-object techniques prohibitively time consuming. Instead, we have implemented and tested two of the existing approaches to 3-D MOS, a brush and a lasso, as well as a new technique, a magic wand, which automatically selects objects based on local proximity to other objects. In a formal user evaluation, we have studied how the performance of the MOS tools are affected by the geometric configuration of the objects to be selected. Our studies demonstrate that the performance of MOS techniques is very significantly affected by the geometric scenario facing the user. Furthermore, we demonstrate that a good match between MOS tool shape and the geometric configuration is not always preferable, if the applied tool is complex to use.

1 Introduction

Selection of multiple objects (MOS) is a frequent goal of user interactions in desktop environments. The prime example of this is the ubiquitous selection box used for picking multiple icons located in a contiguous region on a computer desktop. Many image processing applications, such as Adobe Photoshop or GIMP, include tools for selecting a large number of pixels to be the target of further processing. Furthermore, MOS is very often used in computer games, where the player has to efficiently assign the same commands to several characters or units. All of the mentioned tasks, and the techniques used to accomplish them, can collectively be referred to as 2-D MOS, since they are carried out in a 2-D context.

Once MOS moves beyond the 2-D desktop and into 3-D, the case becomes more complex, and fewer studies exist. The added complexity comes from several sources: More degrees of freedom to control, a lack of standardized MOS tools, and the possibility of occlusion. The lack of studies on 3-D MOS cannot be explained by a lack of potentially useful applications of 3-D MOS. To illustrate this, we give two examples:

1) Architectural visualizations [4, 17] are often populated by many repetitions of the same objects, e.g. windows, doors, pillars, furniture etc. Efficient manipulation of entire classes of such objects, or other subsets of the avaialble objects, relies on being able to efficiently select them in the first place. 2) In data visualizations, there are often thousands of glyphs or voxels floating around in 3-D

space [14], each representing a sample of the visualized database. In many cases, it is useful for a user exploring such a visualization to be able to highlight groups of glyphs or voxels that are of particular interest [5], or to be able to add annotations [7] to specific parts of the visualization. Once again the prerequisite operation is a MOS task.

A main distinguishing feature of the two examples given above is in the number of target objects to be selected, which we henceforth refer to as the *scale* of the MOS task. In an architectural visualization, serialization of single-object selection techniques may be adequate and efficient, since the number of target objects is often manageable. I.e. it will be rare to need to simultaneously manipulate thousands, or just hundreds, of objects in the same virtual room or building. For this reason, MOS for architectural visualizations will most likely be on a small scale in typical scenarios. We henceforth refer to small-scale MOS as SS-MOS.

However, in the case of data mining applications, the scale of the MOS increases drastically, since it is not uncommon to work with databases consisting of thousands of samples. Thus, this application is an example of large-scale MOS (LS-MOS). Since serialization of single-object techniques unavoidably becomes more and more impractical and time-consuming to use as the task scale increases - e.g. imagine pointing out 1,000 objects on a one-by-one basis - the techniques applicable to LS-MOS tasks are potentially quite different from those applicable to SS-MOS. This makes the technical challenge of designing good LS-MOS techniques interesting. Furthermore, efficient LS-MOS techniques in 3-D are currently relatively unexplored. For these reasons, this paper only deals with MOS tasks, where the scale of the task is at least well into the hundreds.

Thus, the main point of this paper is to test and evaluate the usefulness and performance of different 3-D LS-MOS tools. Furthermore, we seek to evaluate how the geometric layout of the objects to be selected affects the efficiency, precision and ease-of-use of the tools. The results gained through this study are therefore useful to future designers of 3-D LS-MOS toolboxes, since they provide information about the trade-offs made when using the tools. That is, recommendations of when to use specific tools, and, just as importantly, when to avoid them.

2 Related Work and MOS Theory

The tasks that users perform while immersed inside virtual environments are traditionally split into four categories: Selection, manipulation, navigation, and system control. This distinction of categories is e.g. presented by Bowman et al in [2], where they present a comprehensive design space for 3-D selection techniques. However, the work only mentions single-object selection techniques. The presented design space for selection tasks includes a category for automatic selection, but does not present any examples of such techniques.

Although MOS tasks clearly belong within the well-established selection category, 3-D MOS, and in particular 3-D LS-MOS, does not appear to have been the subject of much study. A few exceptions are presented in [11, 22, 23]. For this reason, much of the related work concerns traditional single-object selection.

2.1 Single-Object Selection

For several reasons, single-object selection (SOS) techniques are of interest in the design of 3-D MOS techniques. First of all, any SOS technique has the potential to be used as a MOS technique, if used serially. However, serialization of SOS becomes prohibitively time-consuming as the scale of the MOS task grows. Secondly, many SOS techniques are inherently MOS techniques with an added disambiguation step. This step is included in order to pick just one of the candidate objects that fall within a selection volume. The inclusion of a disambiguation step modifies a MOS technique into a SOS counterpart. As such, many SOS techniques hold additional potential as MOS techniques by removing or modifying the disambiguation step.

Ray casting is one of the most well-established SOS techniques [13, 1]. In ray casting, the selection volume is a half-line or a very narrow cylinder extending from the user's hand infinitely into 3-D space like the beam of a laser pointer. In many cases, the selection ray intersects more than one object, which calls for the inclusion of a disambiguation step. The potential of ray casting in itself as a non-serialized 3-D LS-MOS technique is limited, however, since the selection volume is very small. This issue may be alleviated by using the selection ray to pick out a single object, after which all similar or nearby objects are automatically included in the selection. The similarity/proximity criterion can be arbitrarily chosen. The idea of automatically expanding the selection from a single object forms the basis of the *magic wand* technique which is introduced and evaluated in this paper (see section 3.3).

Variations on basic ray casting exist where the ray is replaced by a cone, thus creating a flashlight metaphor. Since cones have larger volumes than rays, they hold more potential as MOS tools in their own right. The cone approach to SOS is e.g. used in [10], where disambiguation is performed by picking the object closest to the axis of the cone. Another example of a cone-based selection technique, called shadow-cone selection, is introduced in [20], where the disambiguation mechanism is based on sweeping the cone in space around the desired object. Objects that stay inside the cone throughout the sweep are selected. As such, shadow cone selection both works in SOS and MOS contexts in its original form. A similar ray-based method, where objects intersected by a ray receives selection scores based on the temporal profile of the ray motion vs. the intersected objects is presented as IntenSelect in [3].

The shadow cone selection is also discussed in the light of a general model of selection presented by Steed in [19]. This model is very general, and as such, the model proposed in that paper covers all volume-based SOS and MOS techniques. Non-volume-based techniques, however, are not covered by the proposed model. I.e. tools that rely on various forms of indirect selection, e.g. selection from a list of object names, or automated selection, which is one of the techniques evaluated in this paper.

2.2 Multiple Object Selection

2.2.1 2-D MOS

In 2-D desktop contexts, MOS tasks are very frequent. As such, the techniques for solving the problem in 2-D are well-established. Wills presents a comprehensive design space for 2-D MOS techniques in [27]. In that paper, a distinction between *brush*-type techniques and *lasso*-type techniques is made. This distinction is also valid for 3-D MOS.

Brushes: With a brush, the selection is made inside a persistent object, called a brush, which the user can manipulate in various ways. One typical manipulation is to drag the brush around. It is possible to add more to the selection simply by moving the brush while making an indication to select (e.g. clicking a button). The brush-metaphor is very clear and intuitive, because dragging the virtual brush around is very similar to painting with a real brush.

It is fairly straightforward to adapt the idea of a selection brush to 3-D MOS, since all that is needed is to 1) use a 3-D shape instead of 2-D shape for the brush and 2) map the controls of the brush to a motion-tracked 3-D interaction device. In this paper, we have chosen to use a spherical 3-D brush with adjustable radius as one of the three evaluated MOS techniques.

Lassos: The lasso category of selection techniques is based on the user defining a temporary selection shape, called a lasso. All objects that lie within the created lasso are selected, after which the lasso disappears. If the user wishes to expand the selection, more lassoes must be defined. In 2-D, a lasso can be made in several ways. Desktop selection rectangles are a well-known example of such a method. In other applications, lassos are made by tracing out a closed, free-form contour on the screen using a mouse.

The lasso concept can also be adapted for use in 3-D. However, the case is not as straightforward as with the brush, because the steps involved in creating the actual lasso are non-trivial to convert from 2-D. First of all, the lasso shape must be a closed 3-D volume, or at least an infinite extrusion of a 2-D shape. This means that the desired shape of a 3-D lasso is probably the primary design choice to make. It only becomes possible to design an efficient way of producing the lasso, after deciding on the class of 3-D shape. Efficient creation of 3-D shapes, specifically 3-D boxes - the natural 3-D extension of 2-D rectangles, is discussed in [23, 21]. A box-shaped lasso, created according to the 3C technique introduced in [21] is the second 3D LS-MOS technique evaluated in this paper.

Automatic techniques: The work of Wills [27] does not account for the possibility of automating part of the MOS process, although the proposed design parameters do include the possibility of modifying the selection volume based on the objects inside of it. Nonetheless, automatic 2-D MOS selection tools are very common in desktop applications, especially in image processing. Such tools use region growing algorithms to expand a selection to all similar pixels connected to an initial seed pixel indicated by the user. The similarity of pixels is typically based on their RGB colour values. This is exactly the scheme which we have chosen to evaluate as the third way of performing MOS tasks. In Adobe Photoshop, the automatic selection tool is named a magic wand. This has inspired us

to use the same name for our automatic tool in order to give users a metaphor to relate to. However, the region growing method underlying our magic wand is somewhat different from the one found in Photoshop. See section 3.3 for more details.

2.2.2 3-D MOS

Few studies directly concerning 3-D MOS currently exist. In one of the currently most comprehensive studies on the subject, Lucas et al [11] presents a design space for 3-D MOS techniques. This design space identifies 6 parameters to consider in 3-D MOS design. Subsequently, variations of two of these parameters, concurrency and spatial context, are used in designing and evaluating four different 3-D MOS techniques. One technique was serialized ray casting. Two of the techniques were performed through a 2-D view of the scene on a hand-held tablet, and as such, are more or less clones of well known desktop MOS techniques. The final technique featured a persistent selection box, i.e. a box-shaped brush, which could be scaled, rotated, and positioned arbitrarily through a combination of two techniques: Go-Go [15] and PORT [12]. The box selection technique is the only one of the four techniques tested by Lucas et al which is both compatible with the LS-MOS context of this paper, and with the non-see-through HMD setup used. The number of target objects was in the range from 9 to 64, thus staying reasonably within the realm of SS-MOS, where serialization of SOS techniques is viable. The objects were laid out in a non-randomized, rectangular grid pattern in all cases. Furthermore, the selection task was visualized on a projection screen. These facts differentiate the study of Lucas et al from the one in this paper.

In the work of Ulinski et al [23], the subject of study is MOS using boxes of arbitrary position, orientation, and scale. The boxes studied were lassos, since they were non-persistent. Several techniques for creating the lassos were evaluated wrt. two binary factors, density and occlusion. The studies were carried out by users interacting with a monoscopic rendering of the data displayed on a tabletop monitor. The spatial layout of the target objects were box-shaped in all cases, meaning that no test of the influence of layout, beyond the mentioned factors, was made. I.e. there were no cases where the target objects' spatial layout did not match the shape of the lasso. No quantitative information about the scale of the task is given, but it is possible that the task scale approaches the scales tested in this paper. Thus, our work differentiates itself from Ulinski et al's by virtue of testing several different spatial layouts of target objects, by testing other tools than 3-D box lassos, and by using an immersive display such as an HMD.

The case of automatic 3-D MOS has also been considered previously. However, it seems that existing techniques rely on the existence of a relational data structure, e.g. a scene graph, behind the rendered virtual objects. This for example means that selecting a virtual table also selects all the objects resting on the table. Such techniques are e.g. presented in [22, 8, 28]. These methods seem very attractive for virtual worlds populated by recognizable objects such as architectural visualizations or computer games. However, these techniques are not applicable in the general case, where the visualized objects come from a database

Figure C.1: The mice used in the experiment. Each mouse is fitted with reflective markers to make it detectable by an infrared camera system. The mice are equipped with the three usual mouse buttons as well as a scroll wheel.

without natural structures such as parent-child relations. This is frequently the case in abstract data visualizations.

3 Design of 3-D MOS Techniques

In this section, we present the specific design used for the three tested MOS techniques. Two of these, the brush and the lasso, relate directly to some of the studies mentioned above. The magic wand, however, is new in a 3-D MOS context. Therefore, the algorithm used in the implementation of the automatic aspects of the wand is of particular interest. The choice of the control mappings in all techniques is heavily based on the available controllers: Two wireless presenter mice fitted with markers to make them trackable with 6 degrees-of-freedom (DoF). This implies that the available controls on the devices apart from the motion tracking are three buttons (left, middle, right) and a scroll wheel. The mice are shown in Figure C.1. The designed techniques take advantage of having two controllers either by being bimanual techniques, or by duplicating the same technique to both hands, giving the users a free choice of which hand to use. Furthermore, the two primary concerns addressed as design criteria are that the techniques must be usable in an HMD context, and that they must be applicable as 3-D LS-MOS techniques.

3.1 Brush

The two primary decisions to make when designing a 3-D brush is the shape of the brush and its control mappings.

In this study, we have chosen to use a spherical brush. The main motivations for this choice are twofold. First of all, a sphere is a well-known shape which should be easy to control for the average user. The number of DoF needed for controlling all parameters of a sphere is relatively low. A spherical brush has a maximum of 4 degrees-of-freedom, i.e. a 3-D position and a radius. This fact is important to this study, where the time available for familiarizing oneself with the tool is fairly limited. Secondly, a spherical brush with adjustable radius should be applicable and relatively precise in many contexts. As such it should provide a very generally applicable tool.

The main disadvantage is expected to be that it will be difficult to select targets located at the edges of a non-spherical volume, e.g. a box shape, where clutter is present close to the targets. E.g. it is difficult to use a sphere to select target objects in the corner of a box surrounded by undesired objects (noise), without also selecting some of the noise.

The choice of a spherical shape also has direct implications on the control scheme used. Since the sphere is a closed volume in space, unlike a ray or a cone, the actual brushing process becomes very local and direct. I.e. the users must place the brush at the desired targets before doing the brushing. This is unlike a ray or a cone, where users can aim and select at a distance. We decided to go with a sphere and the resulting control scheme, because a cone is impractical in scenarios where noise is obstructing a clear path of pointing. This is the case in many of the scenarios we have tested.

For the same reasons, we have also chosen to go with the same control scheme for the other MOS techniques. This not only has the mentioned benefit of working well in cluttered scenarios. It also reduces the amount of different interaction schemes that the test subjects will have to learn. Furthermore, it puts the tested techniques on even ground in the evaluation, meaning that the control scheme is not a factor of the study.

Thus, the controls are a direct mapping of the positions of the user's two hands to the positions of two virtual brush spheres, based on the information provided by a motion tracking system. The brush spheres are made semi-transparent in order to allow better perception of the objects inside the brush. This design choice is in line with previous work by Zhai et al [29]. The adjustable radius is controlled by scrolling the wheel on the controller. We have chosen to bound the range of allowed radii to avoid excessively small or large brushes. The control of indicating that the objects currently inside the brush should be committed to the selection set is mapped to the left buttons. I.e. clicking the left button once selects all objects inside the brush, while holding the left button allows the user to keep selecting objects while sweeping the brush through the volume of objects. The brushes in action are shown in Figure C.2.A and Figure C.2.E.

In order to allow the user to correct mistakes, clicking the right button toggles from the basic selection mode to a deselection mode, where the brushing removes objects from the selection set. The current mode of operation is indicated by the colour of the brushes. This mode toggling feature is also available for the lasso, whereas deselection works differently with the magic wand.

The user can select or deselect objects by repeating the process as many times as desired. Thus, the designed techniques are examples of parallel 3-D MOS tools in the terminology of [11], which can be used in a serial fashion to refine or expand a selection.

3.2 Lasso

For the lasso metaphor, we chose to use a 3-D version of the well-known 2-D selection box lasso from the desktop. Since an arbitrary 3-D box has 9 degrees of freedom, the construction technique for the lasso has to comply with this fact. Control of all 9 degrees of freedom is needed to allow for full control of the flex-

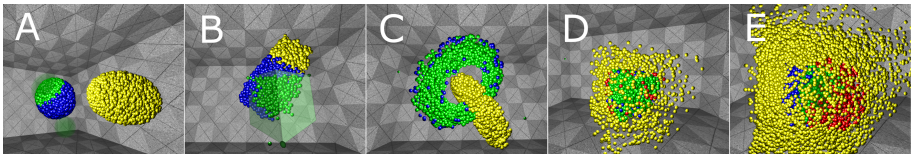

Figure C.2: Screenshots of various experimental conditions and techniques. Blue glyphs are targets, yellow ones are noise. Correctly selected glyphs are highlighted in green, incorrect ones are highlighted in red. (A) The separated clouds scenario with the 3-D spherical brush shown as faint, semi-transparent green spheres. (B) The adjacent clouds scenario with the 3-D box lasso technique in use. (C) The entangled clouds with the magic wand technique in use. The magic wand has been used once to select most of the blue cloud without adjusting its proximity threshold. (D) The embedded nucleus scenario with the magic wand as selection tool. As in part C, the wand has been used once here to make a selection. A few false positives are showing as red glyphs. (E) The uniform embedding scenario with a large 3-D brush in use. This scenario is particularly difficult due to the large amount of noise occlusion, and the close proximity of the noise and target glyphs.

ibility offered by the box shape. This makes the technique somewhat more difficult to use than the 4 DoF spherical brush. However, it also has the potential to be a better match for the presented cloud of target objects, if the cloud has angular corners or planar surfaces. Controlling 9 DoF in an efficient manner is no simple matter. For this reason, we have chosen to implement the existing technique, 3-corners (3C), which provides the greatest degree of precision according to [21].

The 3C technique requires the user to indicate the positions of 3 corners on the desired box in a specific pattern. We have chosen to go with the variant of 3C named 3+3+3 DoF 3C in [21]. This suits the direct interaction pattern chosen for the brush technique well, since it is simple to point out specific points in space, in this case box corners, in a direct way. I.e. the user has to place a hand at a desired corner position, after which a click places the corner there. The hand positions are indicated by small spheres. After making three clicks, the lasso is completed, and all objects inside the lasso are selected. The user can interactively see the lasso currently resulting from the positions of the hands as soon as the first lasso corner has been placed. As was the case with the brush, the lassos are rendered as semi-transparent boxes. An example of a box lasso in use is shown in Figure C.2.B.

As was the case with the brush, the user has the option to right-click in order to toggle to a deselection mode, where the boxes removes objects from the selection set. Similarly, the user is allowed to make as many selection/deselection boxes as desired to reach the end result.

3.3 Magic Wand

The magic wand is based on its namesake technique used in 2-D image processing applications. The main idea is that the user indicates a single object, the seed,

which is representative of the objects in the desired selection. An algorithm then takes care of expanding the selection from the seed to all objects which are similar enough to the seed according to some criterion.

In general terms, any selection technique, SOS or MOS - automatic or manual, can be viewed as a binary classification task, where the objects of the scene are split into two clusters, the target and the noise. In the special case of SOS, the target cluster only contains one object. Thus, the type of algorithm needed for the magic wand belongs to the category of clustering algorithms.

Many clustering algorithms are based on knowing the expected spatial distribution of the objects to be clustered. However, in the general case of MOS, nothing is known a priori about such distributions. I.e. the spatial layout of the objects can vary wildly from case to case. Therefore, we have aimed to design a magic wand algorithm which does not make any assumptions about the overall shape of the clouds to be selected.

Instead, the assumption that we make is based on human perception: People tend to think of densely packed groups of objects as a whole instead of invidual constituents. This is e.g. a well-established fact in the gestalt laws of human visual perception [26]. The gestalt law of proximity is of special interest here, since it provides a proven theoretical background to base the automatic techniques on. As such, our magic wand technique is designed to select all objects that feature high proximity to other objects in the cluster, and ultimately to the seed. Furthermore, to be applicable as an interaction technique, the clustering algorithm should not be so computationally expensive that it prevents the clustering from happening in real-time with the scales of MOS tasks used in our study.

Such a procedure is e.g. presented as the initial grouping step of the clustering algorithm introduced in [6]. There, the criterion used for growing a cluster is to always add the candidate which is closest to *any* current member of the cluster. In our case, where we want to base the algorithm on the gestalt law of proximity, the closeness of cluster members is based on Euclidean distance in 3-D. More data dimensions can be taken into account in the distance measurement, if desired, to make the clustering sensitive to other cues than spatial proximity.

It is worth noting that the use of Euclidean distance as proximity criterion does not necessarily lead to spherical clusters: When viewed on a global level, it is not the Euclidean distance from the seed to the other members of the cluster that matters. Instead, all members have in common that there is at least one other member located somewhere within local proximity. I.e. clusters can have members very far from the seed, as long as there exists a path of sufficiently small jumps through the cluster from the seed to the distant cluster members. This principle maintains the idea that the selected objects should be in a region of similar density/proximity between the objects.

The clustering principle that a single neighbour inside the cluster is enough to add a candidate to the cluster is known as *single-linkage clustering*. An optimally efficient $O(n^2)$ single-linkage algorithm known as SLINK is presented in [18]. However, in our case we can improve the situation somewhat, since a large part of the algorithm, i.e. all the pair-wise distance computations between objects, can be pre-computed once the objects are initialized. This is true as long as the objects do not move relative to each other. Also, all inter-object connections which

are longer than a set acceptable threshold can be pruned after the inter-object distance table is known. This removes the need for starting the clustering from scratch when the user indicates a new seed. Furthermore, we are only interested in the cluster that contains the seed, not all the clusters that are present in the data set.

Mapping the above to user controls has the following implications:

1. The user must be able to select the seed object.
2. The user must be able to adjust the proximity threshold used when deciding which objects belong to the selection set.

Pointing out the seed is done directly using a small, spherical cursor - just like the indication of the corners of the 3-D box lasso. Furthermore, adjustment of the maximum allowed distance between cluster members and their closest neighbour in the cluster is mapped to the scroll wheel. This means that our magic wand is a 4 DoF tool. Increasing the threshold value has the effect of lowering the requirements of cluster membership, i.e. expanding the selection. The opposite is true when decreasing the threshold value. At the minimum value of 0, only the seed is admitted into the cluster, effectively making the magic wand a simple SOS tool. Figure C.2.C and C.2.D show the the magic wand being used.

Thus, selection and deselection with the magic wand happens when the scroll wheel is used. As with the brush and the lasso, the user is allowed to add to the selection by adding new seeds. This implies that in some difficult cases, the user has the option to keep the proximity threshold at a low level, in order to serially add new seeds with small clusters to the selection set without much risk of accidentally selecting noise objects.

There are many possible modifications to the basic magic wand technique. Some examples are: Including an upper limit on the size of the selection set, limiting the number of allowed jumps from the seed, increasing the linkage requirement from a single neighbour to more neighbours, etc. However, in this study we have chosen to go with the basic scheme as outlined above.

4 Experiment

With the three MOS techniques outlined and motivated, the performed experiment can be explained.

4.1 Geometric Scenarios

The main purpose of the experiment is to test how the three chosen MOS techniques fare in scenarios of different geometric layout. In choosing scenarios, we have aimed to include some that represent a broad range of all possible scenarios, and to design the scenarios such that they potentially hightlight the strengths and weaknesses of each technique. As such, scenarios encountered in real applications should conceptually match a combination of those tested in this study. A geometric scenario consists of two randomized clouds of glyphs: A target cloud

rendered as blue glyphs, and a noise cloud rendered in yellow. The glyphs temporarily change colours to green (target) or red (noise) upon entering a selection volume, and permanently so upon being committed to the selection set. Examples of all 5 chosen scenarios are shown in Figure C.2.

The randomization is performed such that the overall shape of the clouds adheres to the design of the scenario. Both clouds are typically created with an average density of $50,000$ glyphs/m^3, which makes it straightforward to perceive the overall shape of the clouds from the visualized glyphs. The chosen density implied that the mean number of target glyphs was app. 1700, while the mean amount of noise glyphs was app. 3300. The glyphs were rendered as Phong-shaded, spherical point sprites in the experiment. The goal of the user task was to select all target glyphs while avoiding selection of the noise glyphs. The best possible result, i.e. one where all targets are selected without any of the noise glyphs being selected, is referred to as a perfect selection.

4.1.1 Separated Clouds (SC)

The separated clouds scenario is the baseline best-case scenario, which is expected to be straightforward for all techniques. It consists of two spherical clouds, which are spatially well-separated. As such it should be simple to make a perfect selection no matter which technique is used.

4.1.2 Adjacent Clouds (AC)

In the adjacent clouds scenario, two box-shaped clouds are placed adjacent to each other with only a very small separation distance. This layout poses considerable difficulty for the magic wand, because the local proximity criterion can easily cause the clustering algorithm to bridge the gap between the two clouds. Furthermore, the spherical brush can be difficult to use in the region close to the boundary between the two clouds without accidentally selecting some of the noise glyphs. The box lasso shape is a perfect fit for this scenario.

4.1.3 Entangled Clouds (EC)

In this scenario, the clouds are shaped as two tori. The tori are oriented such that the plane of one is perpendicular to that of the other. Furthermore, they are offset from each other such that one runs through the centre of the other. The two tori are well separated everywhere. This is a perfect case for the magic wand, since the local proximity criterion allows the clustering to walk all around one torus without ever jumping to the other one. Using the brush is also expected to be somewhat straightforward. A skilled user should be able to adjust the brush radius once, and then paint the entire target torus using a few, precise strokes. The entangled clouds are problematic with box shaped lassos. It is impossible to achieve perfect selection using a single box, since any volume completely containing one of the tori also contains part of the other torus. This means that the user must either employ a serial selection-deselection scheme, or select the target torus a slice at a time.

4.1.4 Embedded Nucleus (EN)

In the target nucleus scenario, the target is presented as a dense nucleus completely embedded in a more sparse cloud of noise. There is no noise inside the volume spanned by the target nucleus. Both the target and the noise clouds are box-shaped. This scenario is expected to be straightforward with the magic wand, since it essentially clusters points based on density. This allows the automatic selection to reach good results with only a few false positives within a short amount of time. The brush will probably have problems avoiding false positives, and will likely begin to suffer from occlusion problems, where the noise gets in the way of properly controlling the brush. The box lasso perfectly matches the task, however, visual occlusion problems may degrade its precision.

4.1.5 Uniform Embedding (UE)

The UE scenario is almost identical to the EN scenario. The only difference is that the noise and the target are equally dense. The UE scenario is a worst-case scenario. The only possible worse situation would be the case, where the target and noise volumes are overlapping in space. None of the techniques are expected to perform well here, however, a uniform embedding is a particularly challenging scenario for the magic wand and the brush. The box lasso has the potential to make a perfect selection in just one selection operation, however, this requires that the user is not hindered too much by the amount of visual occlusion present.

4.2 Design Matrix

Having 5 geometric scenarios and 3 different MOS techniques, produces a total of 15 different conditions to make up the 2-factorial randomized complete block design, which we have chosen to use. All conditions are outlined in Table C.1.

Table C.1: All combinations of the two factors of the experiment. The numbers in the table will be used to refer to the specific combinations of the two factors in the analysis of the results. The (+), (.) and (-) labels indicate if the technique is expected to perform well (+), average (.), or badly (-) in the given scenario.

Technique	Scenario				
	SC	AC	EC	EN	UE
Lasso	0 (+)	1 (+)	2 (-)	3 (+)	4 (-)
Brush	5 (+)	6 (.)	7 (+)	8 (.)	9 (-)
Wand	10 (+)	11 (-)	12 (+)	13 (+)	14 (-)

4.3 Hypotheses

Based on the preceding analysis of MOS techniques, and the presentation of the scenarios above, the hypotheses of the experiment are:

H1 The combination of MOS tool and geometric scenario significantly affects the selection performance.

H2 The box lasso is better than the other techniques in the adjacent clouds (AC) scenario.

H3 The magic wand is better than the other techniques in the entangled clouds (EC) and the embedded nucleus (EN) scenarios.

H4 Overall, the magic wand is faster to use than the other techniques.

H5 Overall, the brush is easier to use than the other techniques.

4.4 Response Variables

The notion of performance can be measured in several ways in for a MOS task. There are three main categories of response variables, which are relevant to the hypotheses: Completion time, selection quality, and ease-of-use. While the completion time of a task is straightforward to measure quantitatively, the other two categories offer several options.

The most basic approach to measuring MOS performance is to count the number of true positives (TP), false negatives (FN), true negatives (TN), and false positives (FP). The goal of any MOS task is to maximize the TP and TN counts while minimizing the FP and FN counts.

A common approach to quantitatively expressing the quality of a binary classification/selection is through the concepts of sensitivity and specificity. These quantities are derived from the basic counts of the four mentioned categories. The sensitivity is the amount of targets selected out of the total number of targets. Conversely, the specificity expresses how many of the noise objects have been correctly avoided relative to the total amount of noise. Both quantities are typically measured in percent. As such, the perfect solution to a MOS task reaches 100% sensitivity while maintaining 100% specificity. One of the advantages of using sensitivity and specificity over the raw counts is that it is independent of task scale. I.e. the range of possible sensitivities and specificities is the same regardless of the absolute counts.

There are many other possible ratios, which can be computed from the basic counts, but to avoid complicating the analysis, we have chosen to only work only with sensitivity and specificity to express selection quality.

The concept of ease-of-use is a subjective assessment. This assessment can be given in a multitude of ways, e.g. through post-test questionnaires, structured interviews, informal discussions, etc. In order to facilitate statistical analysis along with the other response variables, we have chosen a quantitative approach, where participants are asked to subjectively quantify their perception of task difficulty on a 1 (trivial) to 10 (impossible) scale after completing the trials of each test condition. The specific question asked was “How difficult do you think it was to solve the task well?”. To supplement the subjective measurement, we also decided to count the number of operations used in each trial to get an objective measurement of the ease-of-use. Operations were counted through the number of

selection/deselection indications (i.e. lasso completions, brush strokes, or magic wand seed selections), and adjustments made on the scroll wheels.

4.5 Experimental Procedure & Equipment

The experiment was run on an Intel Core i7-2600 3.4 GHz PC with 8 GB of memory and an nVidia GeForce GTX 590 graphics card. The experimental software was a custom made OpenGL renderer running under 64-bit Microsoft Windows 7. For motion tracking, a 24 camera OptiTrack system was used, which allowed unrestricted user motion inside a 2.25 m radius. The HMD was an nVisor SX111 featuring a $102 \times 64^\circ$ total field-of-view at a resolution of 1280×1024 pixels per eye. Two wireless presenter mice fitted with reflective tracking markers were used as interaction devices (see Figure C.1).

In the experiment, the participants first received an introduction to the equipment. Then a few basic, demographic questions (age, gender, 3DUI experience) followed. All participants were instructed that they could stop the experiment at any time, and that they could request breaks.

After donning the HMD, the participants were presented with a few practice scenarios to familiarize themselves with all of the techniques and controls. These practice scenarios were all of the SC and AC types. Once the subjects were comfortable with all three techniques, the experiment commenced. The participants were instructed to select all of the blue glyphs, and to avoid selecting the yellow glyphs *as well as possible* in all scenarios. Not requiring perfect selections was a necessity, since far from all combinations of techniques and scenarios would be perfectly solvable, at least within reasonable time. Furthermore, this approach means that the quality parameters of sensitivity and specificity become meaningful quantities to measure, since they are not always at 100%. The subjects were instructed to let the experimenter know as soon as they felt that they were done with a trial. The experimenter would then press a button, the completion time would be logged, and the experiment proceeded to the next trial.

The sequence of the test conditions was randomized for each subject. Furthermore, all test conditions were repeated 3 times for all subjects. Thus, each subject went through a total of 45 trials during the experiment, which was doable for most subjects in less than an hour. At the end of the experiment, an informal debriefing was made.

5 Results

5.1 Participants

A total of 18 people, 16 males and 2 females, participated in the study. All of the participants were recruited among local university staff and students. The mean age was 27 years ($\sigma = 6.55$ years), and the median self-reported 3DUI experience level on a 1 (novice) to 5 (expert) scale was 3. No payment was offered, apart from some very light refreshments during the experiment.

5.2 Analysis of Hypotheses

All analyses have been made using the statistical software package R [16] using a significance level of $\alpha = 0.05$. Before doing a pre-analysis of the data, three extra response variables were computed from ratios of the directly measured responses. The three ratios were: 1) the sensitivity/operation count ratio, 2) the operation count/completion time ratio, and 3) the sensitivity/completion time ratio. These ratios provide new insights, which are not detectable in the original responses, i.e. 1) the amount of sensitivity gained per operation performed, 2) the speed of each operation, and 3) the sensitivity gained per time unit.

A pre-analysis of the data was carried out to check if it conformed to the requirements of ANOVA analysis. The assumption of independence is satisfied by the fact that independent subjects were used. The normality of the residuals was checked using Q-Q plots. No big problems were detected, however, the sensitivity, specificity, and operation count responses all have some outlier values in their Q-Q plots. The homoscedasticity requirement was not fulfilled for any of the response variables, and attempts to correct this through the use of variance stabilizing transformations failed. We did, however, observe that a logarithm transformation on all of the response variables involving completion time, made the histograms conform much better to a normal distribution bell curve. Therefore, we have used $\log$ transformations on all completion time-dependent variables. Interestingly, this fact is in line with other literature on task completion times, which tend to be log-normal distributed [25, 9]. This fact serves as a partial validation of our experimental procedure. To be sure that we are not reaching the wrong conclusions, we verified that a non-parametric Friedman test reaches the same conclusions as ANOVA. For this reason, we chose to use ANOVA in spite of the violation of a basic ANOVA assumption. All pairwise post-hoc tests were carried out using the Tukey's honest significant difference method.

5.2.1 Hypothesis 1

The hypothesis H1 stated that there would be a significant effect of the combination of MOS tool and geometric scenario. This hypothesis is unambiguously supported no matter which response variable is considered. The p-values are all $\ll 0.001$. This means that the choice of MOS tool makes a big difference depending on the geometric scenario facing the user, both in terms of the quality of selection, completion time, and the subjective judgment of ease-of-use.

5.2.2 Hypothesis 2

H2 hypothesized that the box lasso was the best tool to use in the AC scenario. This means that test conditions 1 (lasso), 6 (brush), and 11 (wand) must be compared. The conclusion here is not so clear-cut as with H1. The performance of the box lasso in the AC scenario relative to the other techniques depends on the response variable chosen. Wrt. subjective difficulty, the lasso is deemed significantly easier to use than the magic wand ($p < 0.001$), but there is no significant difference between the brush and the lasso. With respect to completion time,

there are no significant differences among the techniques. In the case of sensitivity, the lasso does outperform the magic wand ($p < 0.001$), but not the brush. The same result is the case wrt. specificity. If the tradeoff ratios are considered, no significant difference is found wrt. the sensitivity achieved per time unit. However, if the sensitivity achieved per operation is considered, the lasso is superior to both the brush and the magic wand (both $p < 0.001$). I.e. the lasso is very good in the adjacent scenario, if judged by the quality it achieves relative to the number of lassos that you have to use. The fact that the lasso is not performing any better is surprising. The main impact of this result is that users may actually not subjectively or objectively prefer to use a tool which is perfectly shaped for the job, if the tool is too complex to use compared to using a simpler tool multiple times. This result is contrary to the speculations on the issue in previous work, e.g. on pp. 20-21 of [24], where it is stated that it is desirable to use a flexible shape that fits the shape of the data. It is likely, however, that extensive training in the usage of the complex tool modifies this result.

5.2.3 Hypothesis 3

This hypothesis deals with the performance of the magic wand in the EC and EN scenarios. The test conditions of interest are 12 and 13 (wand) compared to 2 and 3 (lasso) and 7 and 8 (brush). Specifically, H3 hypothesizes that the magic wand will outperform the other techniques in those scenarios. With respect to subjective difficulty, the magic wand is better than the lasso ($p < 0.001$ and $p = 0.013$). However, there is no significant difference between the wand and the brush. The response variable that really sets the wand apart from the other two techniques is completion time, where the wand is significantly faster to use than any of the others (p-values in the range from 0.023 to < 0.001). There is not much difference wrt. sensitivity and specificity, the only signficant result being that the wand reaches significantly higher sensitivity in the EN scenario. If the sensitivity gained per time unit is inspected, the wand is significantly better than the other two techniques ($p < 0.001$), except when compared to the brush in the EC scenario. The overall conclusion wrt. H3 is therefore that in EC and EN scenarios, the wand mainly outperforms the other techniques wrt. speed, but in terms of selection quality, all techniques achieve similar results.

5.2.4 Hypothesis 4

H4 states that the magic wand is faster than the other techniques in general. The results of H3 already supports H4. Inspecting the completion time response variable across all geometric scenarios, reveals that H4 is supported. The magic wand is significantly faster than the brush ($p < 0.001$), which in turn is significantly faster than the lasso ($p < 0.001$). This is also true, if we inspect the amount of sensitivity gained per time unit using the wand. Here the wand is significantly better than the brush ($p = 0.020$) and the lasso ($p < 0.001$).

5.2.5 Hypothesis 5

In the final hypothesis, we conjectured that the brush would be judged to be

easier to use than the other techniques, viewed across all tested scenarios. The motivation of H5 being true was that the brush featured fewer DoF than the lasso (4 vs. 9), which provides for easier control. At the same time, some of the scenarios were designed to be nearly impossible to do well using the magic wand. Thus, the brush has potential to be a jack-of-all-trades tool, which users would find easy to use. H5 is supported by the data, based on the subjective ratings of difficulty. Thus, the brush is evaluated to be easier to use than the other two tested techniques, $p = 0.026$ (wand) and $p < 0.001$ (lasso). However, if we evaluate ease-of-use in terms of the number of operations needed to accomplish the tasks, then the picture very different. In terms of number of operations, the box lasso uses significantly fewer operations than any of the other two techniques, $p = 0.0026$ (brush) and $p < 0.001$ (wand). This provides even more evidence that users prefer to use a simple tool many times rather than using a better fitting, complex tool a few times.

5.3 Other Findings

One final, interesting point which we noted during the experiments is that there seems to be an asymmetry in the way the participants judge the quality of their selections. It was very noticeable that participants would much rather accept getting even quite large numbers of false negatives (i.e. target objects not selected), than getting a few false positives (i.e. selected noise objects). In terms of the response variables, this implies that the participants were willing to accept a lower sensitivity, if this implied getting a higher specificity. This trend can be statistically tested by comparing the measured sensitivities and specificities in a paired t-test. This test reveals that the average specificity across the entire experiment was higher than the average sensitivity ($p \ll 0.001$). We therefore suggest that future methods for measuring perceived MOS quality, as well as future automated MOS techniques, should recognize this finding. I.e. take into account that false positives are considered worse than false negatives, calling for conservative automatic tools that prioritize maximization of specificity in favour of sensitivity.

6 Conclusion & Perspectives

In this paper, we have presented the following contributions in relation to 3-D MOS tasks. We have presented a thorough analysis of the field of 3-D MOS, including the distinction between the requirements of small and large-scale MOS tasks. We have presented and evaluated a new technique, the magic wand, for partially automating 3-D LS-MOS tasks. Furthermore, we have made a rigourous experiment demonstrating that, overall, it is very important to design or pick MOS tools based on the geometric scenarios facing the user. In this respect, we have found that: 1) The use of a 3-D magic wand is a very fast technique, but also very sensitive to the geometric scenario, making it either very easy or completely impractical to use. 2) The natural 3-D extension of the 2-D rectangular lasso is not preferred by participants over simpler techniques, even when the simpler options are less suitable for the geometric scenario. 3) The 3-D spherical brush is

a good candidate for a general 3-D MOS tool applicable to many scenarios. The best 3-D MOS approach in future applications therefore seems to be to include an ensemble of MOS tools rather than just a single technique.

Overall, we believe that the results of this experiment should be of interest to any future investigators of 3-D MOS tasks, especially those performed on a large scale, i.e. with too many objects to make serial single-object selection practical. Apart from the tested hypotheses, the finding of an asymmetric perception of selection quality is interesting. The fact that participants were more accepting of low sensitivity than low specificity is something which can be incorporated into future MOS studies.

Acknowledgements

The authors wish to thank all the people who participated in this study.

References

[1] D. A. Bowman and L. F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Proceedings of the 1997 symposium on Interactive 3D graphics*, I3D '97, pages 35–ff., New York, NY, USA, 1997. ACM.

[2] D. A. Bowman, E. Kruijff, J. J. LaViola, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley, first edition, 2005.

[3] G. De Haan, M. Koutek, and F. H. Post. IntenSelect: Using Dynamic Object Rating for Assisting 3D Object Selection. In *In Virtual Environments 2005*, volume 2005, pages 201–209, 2005.

[4] P. Frost and P. Warren. Virtual reality used in a collaborative architectural design process. In *Information Visualization, 2000. Proceedings. IEEE International Conference on*, pages 568 –573, 2000.

[5] R. Fuchs, V. Welker, and J. Hornegger. Non-convex polyhedral volume of interest selection. *Computerized Medical Imaging and Graphics*, 34(2):105 – 113, 2010.

[6] E. Gokcay and J. C. Principe. Information theoretic clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:158–171, 2002.

[7] R. Harmon, W. Patterson, W. Ribarsky, and J. Bolter. The virtual annotation system. In *Virtual Reality Annual International Symposium, 1996., Proceedings of the IEEE 1996*, pages 239 –245, 270, mar-3 apr 1996.

[8] J. Jang and J. R. Rossignac. Multiple object selection in pattern hierarchies. Technical report, Georgia Institute of Technology, 2007.

[9] S. Kochhar, S. Mazzocchi, and P. Paritosh. The anatomy of a large-scale human computation engine. In *HCOMP '10: Proceedings of the ACM SIGKDD Workshop on Human Computation*, pages 10–17, New York, NY, USA, 2010. ACM.

[10] J. Liang and M. Green. JDCAD: A highly interactive 3D modeling system. In *Computer and Graphics*, volume 18(4), pages 499–506, 1994.

[11] J. F. Lucas. Design and evaluation of 3d multiple object selection techniques. Master's thesis, Virginia Polytechnic Institute and State University, 2005.

[12] J. F. Lucas, J.-S. Kim, and D. A. Bowman. Resizing beyond widgets: object resizing techniques for immersive virtual environments. In *CHI '05 extended abstracts on Human factors in computing systems*, CHI EA '05, pages 1601–1604, New York, NY, USA, 2005. ACM.

[13] M. Mine. Virtual environment interaction techniques. Technical report, UNC Chapel Hill CS Dept, 1995.

[14] H. R. Nagel. *Exploratory Visual Data Mining in Spatio-Temporal Virtual Reality*. PhD thesis, Faculty of Engineering and Science, Aalborg University, 2005.

[15] I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa. The go-go interaction technique: non-linear mapping for direct manipulation in vr. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*, UIST '96, pages 79–80, New York, NY, USA, 1996. ACM.

[16] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2011. ISBN 3-900051-07-0.

[17] M. Rafferty, D. Aliaga, V. Popescu, and A. Lastra. Images for accelerating architectural walkthroughs. *Computer Graphics and Applications, IEEE*, 18(6):38 –45, nov/dec 1998.

[18] R. Sibson. SLINK: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, Jan. 1973.

[19] A. Steed. Towards a general model for selection in virtual environments. In *3D User Interfaces, 2006. 3DUI 2006. IEEE Symposium on*, pages 103 – 110, march 2006.

[20] A. Steed and C. Parker. 3D Selection Strategies for Head Tracked and Non-Head Tracked Operation of Spatially Immersive Displays. In *8th International Immersive Projection Technology Workshop*, Ames, Iowa, USA, May 2004.

[21] R. Stenholt and C. B. Madsen. Shaping 3-d boxes: A full 9-degree-of-freedom docking experiment. In *Virtual Reality Conference (VR), 2011 IEEE*, pages 103–110, March 2011.

[22] W. Stuerzlinger and G. Smith. Efficient manipulation of object groups in virtual environments. In *Virtual Reality, 2002. Proceedings. IEEE*, pages 251 –258, 2002.

[23] A. Ulinski, C. Zanbaka, Z. Wartell, P. Goolkasian, and L. Hodges. Two handed selection techniques for volumetric data. *3D User Interfaces, 2007. 3DUI '07. IEEE Symposium on*, mar. 2007.

[24] A. C. Ulinski. *Taxonomy and experimental evaluation of two-handed selection techniques for volumetric data*. PhD thesis, University of North Carolina at Charlotte, Charlotte, NC, USA, 2008. Adviser-Hodges, Larry F.

[25] J. Wang, O. Alonso, and P. G. Ipeirotis. Estimating the completion time of crowdsourced tasks using survival analysis models. *Search*, (Csdm):1–38, 2011.

[26] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, second edition, 2004.

[27] G. Wills. Selection: 524,288 ways to say "this is interesting". In *Information Visualization '96, Proceedings IEEE Symposium on*, pages 54 –60, 120, oct 1996.

[28] R. C. Zeleznik, K. P. Herndon, and J. F. Hughes. Sketch: an interface for sketching 3d scenes. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[29] S. Zhai, W. Buxton, and P. Milgram. The "silk cursor": investigating transparency for 3d target acquisition. In *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence*, CHI '94, pages 459–464, New York, NY, USA, 1994. ACM.